

7.
001.642
C 843
C.4
ESCUELA SUPERIOR
POLITECNICA DEL LITORAL

ESCUELA DE CIENCIAS DE LA COMPUTACION

BASE DE DATOS RELACIONAL
POLIREL

VERSION 2.0

Presentada por:



INVENTARIADO
8/12/16

ANA COX VASCONEZ
VERONICA CUCALON ZENCK
AGUSTIN CHUSAN VITERI
FAUSTO JACOME LOPEZ
FERNANDO PAEZ GALARZA
MATILDE PERE YCAZA
GISSELLE SALMON BOWEN

Director:

ING. ANDRES NAVARRO GAZZOLA

Guayaquil-Ecuador

1984

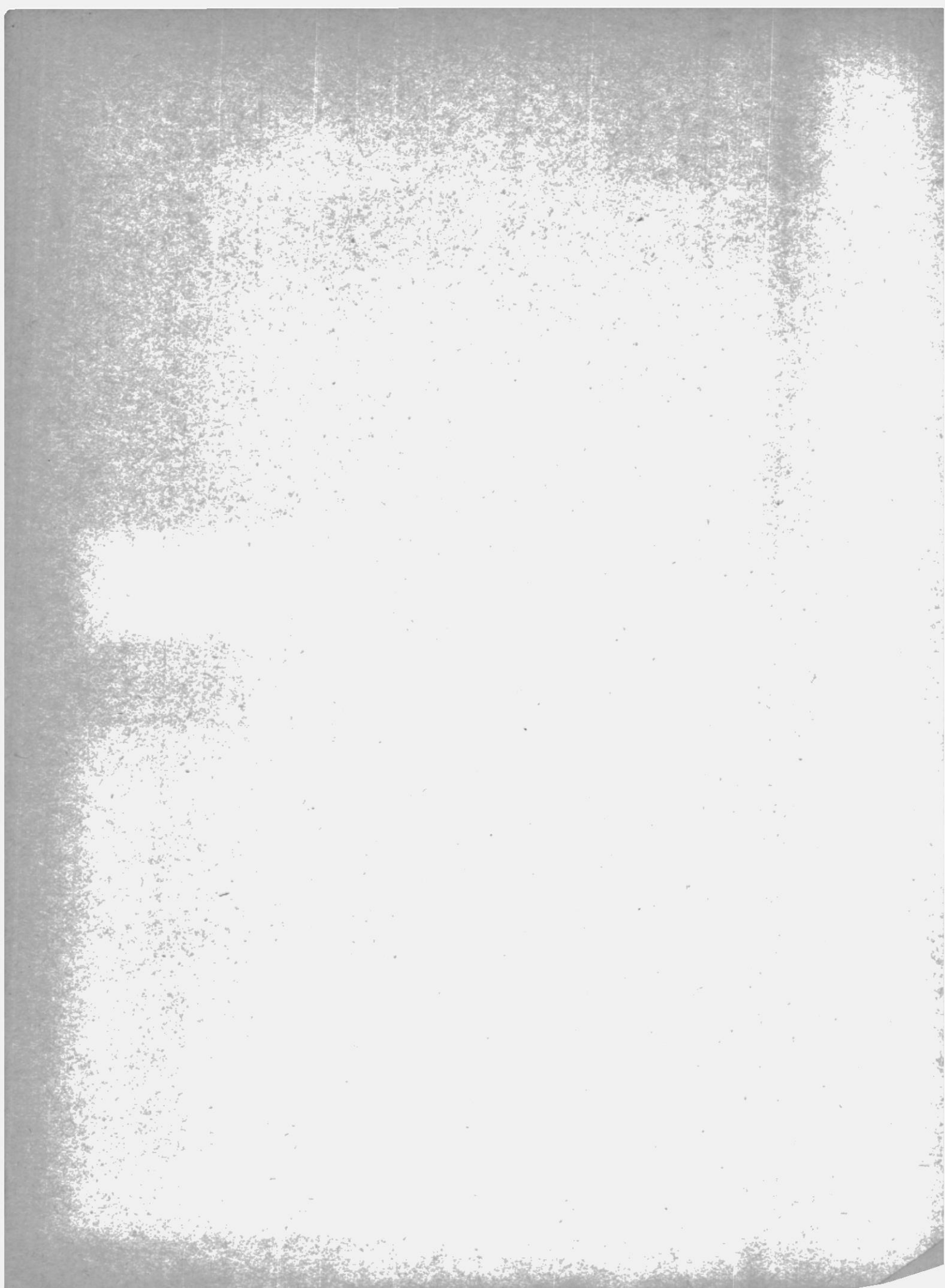
10/10/2012
Inventariado.

11-11-15

Ing. María José Pineda
Asistente de Activos Fijos - CIB



MANUAL DEL USUARIO



PROLOGO A LA VERSION 1.0

La idea de este proyecto se origino ante la necesidad de materializar la teoria y conceptos relacionados con la tecnologia de Base de Datos, y de manera especifica el modelo relacional.

El dictado de este topico durante varios semestres en la catedra de Sistemas de Computo, motivo a la consecucion de un producto terminado, que para propositos practicos y netamente didacticos fue evolucionando hasta convertirse en una realidad satisfactoria.

La intencion al concebir este proyecto fue desde un principio bien clara. No se trataba de desarrollar un producto comercial, sino un sistema que permita al estudiante trabajar y afianzar sus conocimientos que sobre el tema haya sido presentado durante el semestre.

Sin embargo, la concepcion del mismo ha sido efectuada en forma modular y sistematica que bien pudiera servir para un proyecto de investigacion mas ambicioso.

El desarrollo de este proyecto abarca toda una gama de conceptos sobre el area de computacion, no solamente sobre Base de Datos Relacional que es el tema de fondo, sino tambien estructuras de datos, desarrollo de una gramatica sencilla y un procesador para el Query final. Todo esto, teniendo en mente como principal objetivo al usuario, que a

traves de un lenguaje algebraico relacional pudiera en forma interactiva hacer uso en forma sencilla de los diferentes modulos del sistema.

El Sistema consta de una leccion por pantalla, la cual tiene como proposito orientar al usuario en la parte netamente operativa del sistema en conjunto, asi como presentar la teoria pertinente a los fundamentos del Modelo Relacional y el lenguaje algebraico. Otro modulo del Sistema esta constituido por el Editor de Tuplas, el cual permite el ingreso, eliminacion, modificacion y consulta de cualquier tupla en cualquiera de las relaciones creadas por el usuario.

Finalmente, el modulo del Query que a manera de interpretador dinamico permite al usuario formular los diferentes procedimientos y consultas al Sistema Relacional.

NOTA: La direccion de este proyecto, deja por medio de la presente nota constancia del reconocimiento al trabajo de los siguientes alumnos que hicieron posible la Version 1.0: Mirza Arteaga, Xavier Cardenas, Julio Chang, Reinaldo Roca y Maria Mercedes Villacreses.

PROLOGO A LA VERSION 2.0

La nueva version de Polirel es producto del desarrollo y madurez evolutivos del propio sistema, en union con el esfuerzo del grupo de estudiantes que hicieron posible este trabajo.

Polirel en esta version ha sido complementado y mejorado notablemente, no solo en la capacidad del sistema en si, sino tambien en detalles llenos de sutilezas en pro de una mejor aceptacion del producto, manteniendo su ideologia basica y fundamental de ser un Software en principio educacional.

El objetivo primordial de esta version consistio en proporcionar una gran gama de utilitarias que se traduzcan en una mayor flexibilidad en el uso del sistema por parte del usuario, y que ademas, sirva como material de ensenanza integral en la tecnologia de Base de Datos, considerando aspectos de diseno, optimizacion y sintonizacion de las aplicaciones.

Las Utilitarias de Polirel abarcan un gran cantidad de conceptos y propositos practicos, tales como generacion de indices invertidos para optimizacion de las consultas del Query, generacion de indices clasificados para efectos de generacion de reportes, formateo de relaciones, carga y descarga de relaciones, consultas catalogadas, archivo de

auditoria, entre otros.

Como modulo aparte el generador de reportes de Polirel, permite en forma sencilla y versatil la creacion, modificacion de archivos de formatos de reportes para manipulacion de los datos almacenados en Polirel.

Otro de los logros de esta nueva version, es la capacidad operativa de Polirel a traves de un programa aplicativo cualquiera escrito en su lenguaje anfitrión PASCAL. Un usuario puede leer y actualizar cualquier archivo de la Base de Datos utilizando un programa PASCAL y el modulo de interface de Polirel.

La capacidad de almacenamiento de las tuplas ha sido expandida y podra ser seleccionada por el usuario segun requerimientos de cada aplicacion especifica. De igual forma, la implementacion de numeros reales y gran mejoramiento de la presentacion del sistema total, hacen de Polirel un sistema de Base de Datos educativo de gran contenido didactico, asi como de elevado nivel tecnico.

I N D I C E

	Pag.
1. INTRODUCCION	1
2. OBJETIVOS	2
3. REQUERIMIENTOS DEL SISTEMA	3
4. COMO USAR EL SISTEMA	4
5. EDITOR	5
5.1 Definicion de Relaciones	6
5.2 Ingreso/Adicion de Tuplas	10
5.3 Eliminacion de Datos	14
5.4 Correccion de Tuplas	19
5.5 Consulta de Tuplas	24
5.6 Busqueda de Tuplas	30
6. QUERY	35
6.1 Resultados Temporales y Permanentes	38
6.2 Consultas Catalogadas	40
6.3 Operador Seleccion	41
6.4 Operador Join	43
6.5 Operador Proyeccion	45
6.6 Como Obtener Resultados (Comando List/Disp)	47

	Pag.
7. UTILITARIAS	53
7.1 Formateo de Archivos	54
7.2 Generacion de Indices Invertidos	58
7.3 Coleccion de Basura	64
7.4 Clasificacion de Relaciones	66
7.5 Descarga de Relaciones	71
7.6 Carga de Relaciones	73
7.7 Consultas Catalogadas	75
8. ACCESO A LA BASE A TRAVES DE UN PROGRAMA PASCAL	84
8.1 Introduccion	85
8.2 Medio Ambiente de Trabajo	86
8.3 Como iniciar un Programa	88
8.4 Condicion de las Operaciones	89
8.5 Definicion del Formato de Registro de Trabajo	91
8.6 Recuperacion de Datos de la Interface de la Unidad	95
8.7 Programas Demostrativos	99
8.7.1 Programa de Acceso Secuencial a las Relaciones	99
8.7.2 Programa de Acceso Directo a las Relaciones	104
8.7.3 Programa de Acceso Directo a las Relaciones con la utilizacion de Indices Invertidos y de Clasificacion	109

8.7.4 Programa de Actualizacion de Tuplas	114
8.7.5 Programa de Eliminacion de Tuplas	119
8.7.6 Programa de Adicion de Tuplas	125
9. GENERADOR DE REPORTES	130
9.1 Introduccion	131
9.2 Proceso General	132
9.3 Creacion de Formatos	134
9.3.1 Creacion de Datos del Formato General	137
9.3.1.1 Creacion del Formato General de 1 Relacion	137
9.3.1.2 Creacion del Formato General de 2 Relaciones	141
9.3.2 Creacion de Datos de Cabeceras	145
9.3.3 Creacion de Datos de Detalle	152
9.4 Consulta de Formatos	160
9.4.1 Consulta de Formato General	162
9.4.2 Consulta de Datos de Cabeceras	163
9.4.3 Consulta de Datos de Detalle	164
9.5 Correccion de Formatos	165
9.5.1 Correccion de Formato General	167
9.5.2 Correccion de Datos de Cabeceras	168
9.5.3 Correccion de Datos de Detalle	169
9.6 Ejecucion de Formatos	171
9.7 Eliminacion de Formatos	173



10. EJEMPLO

10.1 Pasos a seguir para obtener el Rol de Pagos 177

10.2 Descripción de las Relaciones Definidas 179

10.3 Descripción de Formatos 181

11. RECOMENDACIONES 187

12. APENDICES 188

A) Tabla de Errores 189

B) Tabla de Condición de Operación 191

C) Palabras Reservadas por la Unidad Polirel 193

D) Diagramas Sintácticos 197

1. INTRODUCCION

Este proyecto materializa todos aquellos conceptos adquiridos en las catedras de ESTRUCTURA DE DATOS Y SISTEMAS DE COMPUTO dictadas en la Escuela de Computacion, relacionados con los topicos de BASE DE DATOS. Se ha escogido el modelo Relacional debido a su gran flexibilidad en la manipulacion de datos, ademas de ser un sistema novedoso en conceptos y de facil comprension.

En lo posible se ha tratado que los ejemplos que contiene este manual sean lo suficientemente didacticos para guiar al usuario en el desarrollo de aplicaciones concretas a un nivel de complejidad aceptable.

2. OBJETIVOS

POLIREL ha sido creada con fines academicos, siendo su principal objetivo el de servir como complemento practico a la catedra de Base de Datos, de manera que el estudiante pueda tener una idea clara de ella, sobre todo, en lo que respecta al Modelo Relacional.

Se ha tratado en lo posible de que el sistema sea lo mas sencillo en su uso, generalizando la mayoria de los conceptos ya conocidos sobre el tema.

Sin embargo, el sistema ha sido disenado de tal manera, que bien podria ser utilizado en ciertas aplicaciones.

3. REQUERIMIENTOS DEL SISTEMA

Este Sistema requiere basicamente:

- Memoria de por lo menos 64 Kbytes.
- Dos Disk Drives.
- Tres diskettes que contienen el Sistema de Base de Datos (Base-1, Base-2 y Base-3).
- Un diskette para los archivos de datos.
- El Manual Operativo del Sistema De Base de Datos Polirel.

4. COMO USAR EL SISTEMA

El usuario debera seguir cuidadosamente los siguientes pasos para el uso del Sistema:

- 1.- Inserte diskette Base 1 en el Drive #1.
- 2.- Inserte el diskette de datos, previamente formateado, en el Drive #2.
- 3.- Encienda el Monitor y Computador.
- 4.- Espere el Menu Principal de la Base y Seleccione la opcion deseada.
- 5.- Siga cuidadosamente las instrucciones dadas por el Sistema y obtendra los resultados por usted requeridos.
- 6.- Encienda la impresora si va a obtener algun resultado.
- 7.- Al terminar la sesion saque los diskettes de los Drives, guardelos, apague el monitor, la impresora, si la utilizo y el computador.

5. EDITOR

5.1 DEFINICION DE RELACIONES

Luego de haber creado físicamente los archivos para la relación con la opción FORMATEO DE ARCHIVOS (Ver numeral 7.1), esta opción del Editor permite al usuario definir cuantas columnas tiene una relación y cuales son las características de cada una.

En la primera pantalla que se presenta al ejecutar esta opción (Ver fig. 5.1.1).

Se digita un carácter alfabético o si desea regresar al menú anterior presione la tecla <RETURN>, el carácter identifica la relación a definirse y debe ser el mismo nombre con que fue creada al formatear los archivos, no se debe presionar la tecla <RETURN> luego de digitar el carácter ya que su ingreso es automático.

DEFINICION DE RELACIONES

NOMBRE DE RELACION O <RETURN> -> A

Fig. 5.1.1

En la parte inferior de la pantalla se muestran todos los mensajes por condición de error; por ejemplo si la

relacion no esta en el diskette de datos (Ver fig. 5.1.2).

DEFINICION DE RELACIONES

NOMBRE DE RELACION O <RETURN> -> A

** NO EXISTE RELACION FORMATEADA **

PRESIONE <RETURN> PARA CONTINUAR

Fig. 5.1.2

Si la relacion existe, se ingresa el numero de columnas, la estrategia y el nombre de cada columna con las siguientes restricciones:

- Para el numero de columnas: el numero maximo es diez , luego presione <RETURN>
- Para la estrategia: los caracteres (E=entero, C=caracter, R=real, S=string) dependiendo de la naturaleza de la columna y luego presione <RETURN>
- Para el nombre de columna: la longitud maxima es seis caracteres, sin blancos intermedios, ni caracteres especiales, luego presione <RETURN>.

Todos estos parametros a ingresarse se resumen en la figura 5.1.3.



DEFINICION DE RELACIONES		

NOMBRE DE RELACION O <RETURN> -> A		
DIGITE EL NUMERO DE COLUMNAS -> 4		
COL.	ESTRAT.	NOMBRE
----	-----	-----
1	E	NUMERO
2	S	NOMBRE
3	R	SUELDO
4	C	ESTADO
DESEA CORREGIR? (S/N) ->		

Fig. 5.1.3

Si desea corregir las características de alguna columna o columnas ya ingresadas, se digita la letra 'S' sin presionar la tecla <RETURN>, a continuación se pregunta al usuario que columna desea corregir, aquí se digita un valor entre uno y el número de columnas ingresadas, luego de presionar <RETURN> el cursor se posiciona en la columna escogida; este proceso se repite tantas veces como sea necesario (Ver fig. 5.1.4).

Si los datos ingresados están correctos entonces digite la letra 'N' sin presionar <RETURN>, inmediatamente se mostrará un mensaje que indica el fin de la opción Definición de Relaciones (Ver fig. 5.1.5).

DEFINICION DE RELACIONES

NOMBRE DE RELACION O <RETURN> -> A

DIGITE EL NUMERO DE COLUMNAS -> 4

COL. ---	ESTRAT. -----	NOMBRE -----
1	-	
2	S	NOMBRE
3	R	SUELDO
4	C	ESTADO

DESEA CORREGIR? (S/N) -> S

QUE COLUMNA? -> 1

Fig. 5.1.4

DEFINICION DE RELACIONES

NOMBRE DE RELACION O <RETURN> -> A

DIGITE EL NUMERO DE COLUMNAS -> 4

COL. ---	ESTRAT. -----	NOMBRE -----
1	E	NUMERO
2	S	NOMBRE
3	R	SUELDO
4	C	ESTADO

DESEA CORREGIR? (S/N) -> N

* USTED HA DEFINIDO LA RELACION-> A *

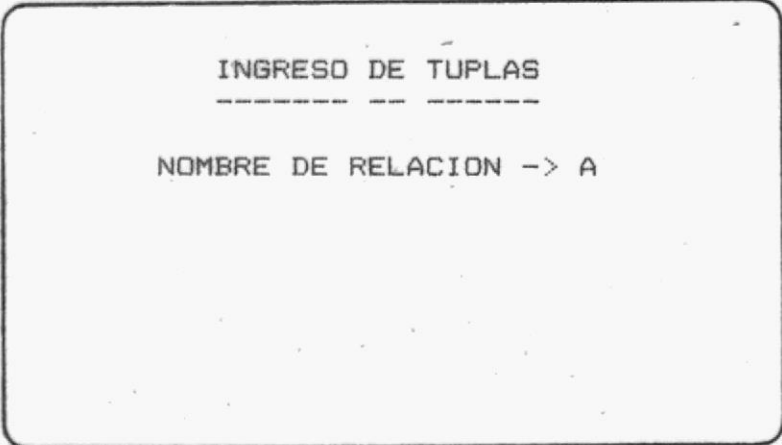
PRESIONE <RETURN> PARA CONTINUAR

Fig. 5.1.5

5.2 INGRESO/ADICION DE TUPLAS

Esta opcion del Editor se realiza solo para relaciones que han sido definidas previamente; permite el ingreso de datos para las tuplas de una relacion dada.

En la primera pantalla que se presenta al ejecutar esta opcion (Ver fig. 5.2.1), se digita un caracter que identifica la relacion.



```
INGRESO DE TUPLAS
-----
NOMBRE DE RELACION -> A
```

Fig. 5.2.1

Luego se pregunta al usuario si desea ingresar una tupla, si digita 'S' aparece en la pantalla el numero de tupla a ingresarse, el nombre y estrategia de cada columna (Ver fig. 5.2.2).

El ingreso de los datos para una tupla depende de la estrategia de cada columna y se realiza de acuerdo a las siguientes restricciones:

- Si es un caracter, se debe digitar caracteres

alfabeticos o numericos

- Si es un dato real, el numero debe tener un maximo de cinco digitos enteros y dos decimales, el punto es opcional. Si digita mas de dos decimales su valor sera truncado
- Si es un dato entero, el numero debe tener maximo cinco digitos enteros y el valor no debe ser mayor a 32767
- Si es una cadena de caracteres, no se permiten caracteres especiales dentro de ella.

INGRESO DE TUPLAS

NOMBRE DE RELACION -> A

DESEA INGRESAR UNA TUPLA (S/N)-> S

NO.TUPLA-> 1

NUMERO(E) -

NOMBRE(S) -

SUELDO(R) -

ESTADO(C) -

Fig. 5.2.2

Un ejemplo de ingreso de datos se resume en la pantalla de la figura 5.2.3.

```
      INGRESO DE TUPLAS
      -----

      NOMBRE DE RELACION -> A

DESEA INGRESAR OTRA TUPLA (S/N)-> _

NO.TUPLA-> 2

NUMERO(E)      2
NOMBRE(S)      RODRIGO PEREZ
SUELDO(R)      6000.00
ESTADO(C)      S
```

Fig. 5.2.3

Para las condiciones de error durante el ingreso de datos, los mensajes respectivos saldrán en la parte inferior de la pantalla (Ver fig. 5.2.4).

```
      INGRESO DE TUPLAS
      -----

      NOMBRE DE RELACION -> A

DESEA INGRESAR OTRA TUPLA (S/N)-> S

NO.TUPLA-> 2

NUMERO(E)      2
NOMBRE(S)      RODRIGO PEREZ
SUELDO(R)      1234567.89
ESTADO(C)

* MAXIMO 5 DIGITOS ENTEROS *
PRESIONE <RETURN> PARA CONTINUAR
```

Fig. 5.2.4

Este proceso se repite tantas veces como sea necesario, y al finalizar el Ingreso de Tuplas se emitira un mensaje que indica el numero de tuplas que ahora contiene la relacion.

```

      INGRESO DE TUPLAS
      -----

      NOMBRE DE RELACION -> A
      DESEA INGRESAR OTRA TUPLA (S/N)-> N
      NO.TUPLA-> 20

      NUMERO(E)      22
      NOMBRE(S)      JUAN PEREZ
      SUELDO(R)      0.00
      ESTADO(C)      D

      LA RELACION TIENE 20 TUPLA(S)
      PRESIONE <RETURN> PARA CONTINUAR
  
```

Fig. 5.2.5

Importante:

Si al tratar de ingresar una tupla aparece en la pantalla uno de los siguientes mensajes:

NO HAY ESPACIO EN ARCHIVO AT

NO HAY ESPACIO EN ARCHIVO ADA

esto significa que la relacion necesita ser reorganizada (Ver UTILITARIAS, Numerales 7.3, 7.5 y 7.6)

5.3 ELIMINACION DE DATOS

El proceso de eliminacion como parte del Editor, es un proceso que sirve al usuario de la Base de Datos POLIREL para el mantenimiento de las relaciones.

Con esta opcion el usuario podra eliminar una o mas tuplas o columnas de una relacion existente.

Es importante mencionar que la eliminacion de una tupla o columna es solamente logica, es decir que estas seguiran fisicamente ocupando el espacio dentro del archivo AT o ADA. El usuario para una mejor utilizacion del espacio dentro de los archivos podra hacer uso de la Coleccion de Basura (Ver numeral 7.3) o de la Descarga y Carga de Relaciones (Ver numeral 7.6), segun sea el caso.

FORMA DE EJECUTAR:

Se presenta la siguiente pantalla:

ELIMINACION DE DATOS
=====

NOMBRE DE LA RELACION O <RETURN> _

Fig.5.3.1

- 1) Si se presiona la tecla <RETURN> retornara al menu del Editor. Si se digita el nombre de una relacion se procede a verificar si existe la relacion, en caso contrario aparecera el mensaje respectivo.

```

      ELIMINACION DE DATOS
      =====
NOMBRE DE LA RELACION O <RETURN> _

      UNO DE LOS ARCHIVOS NO EXISTE
  
```

Fig.5.3.2

- 2) Aparecera el numero de tuplas y el numero de columnas activas que posee la relacion.

Se presenta la siguiente pantalla:

```

      ELIMINACION DE DATOS
      =====
      PROCESO DE LA RELACION  A
      EXISTEN 20 TUPLAS Y 3 COLUMNAS
      ELIMINA TUPLA(T) O COLUMNA(C)? _
  
```

Fig.5.3.3

- 3) Si la opcion es la de eliminacion de tuplas se pedira

al usuario:

ELIMINACION DE DATOS

===== == =====

PROCESO DE ELIMINACION DE TUPLAS

DIGITE NUMERO DE TUPLA--> _

Fig.5.3.4

Digitado el numero de la tupla logica y presionando <RETURN> se despliegan los datos de dicha tupla y aparecera la alternativa:

<< DESEA ELIMINAR >>

Se digitara la tecla 'S' si se desea eliminar y la 'N' en caso contrario.

```

      ELIMINATION DE TUPLES
      =====

      PROCESO DE ELIMINATION DE TUPLAS
      DIGITE NUMERO DE TUPLA      1

      NRO ES: COLUM

      1 (E) CODIGO 1001
      2 (S) NOMBRE AGUSTIN CHUSAN
      3 (R) SUELDO 12000.00

      << DESEA ELIMINAR      S
      ** ELIMINACION DE TUPLA EXITOSA **

      < DESEA CONTINUAR >
  
```

Fig.5.3.5

- 4) Si la opción es la de eliminación de columnas, se presenta todos los nombres de las columnas activas y eliminadas, con sus respectivas estrategias y posiciones físicas.

Se pedirá al usuario que marque digitando con una 'x' las columnas activas que desee eliminar; una vez seleccionadas deberá digitar '0' para finalizar la selección (ver Fig. 5.3.6)



ELIMINACION DE COLUMNAS			
=====			
PROCESO DE LA RELACION A			
NRO	EST	NOMBRE	MARCA
1-	(E)	-CODIGO	
2-	(S)	-NOMBRE	
3-	(R)	-SUELDO	
4-	(S)	-DIRECC	X
<< MARQUE CON X PARA ELIMINAR >> << CON << Q >> PARA SALIR >>			
<< DESEA ELIMINAR >> S ELIMINACION DE COLUMNA EXITOSA			
DESEA CONTINUAR			

Fig.5.3.6

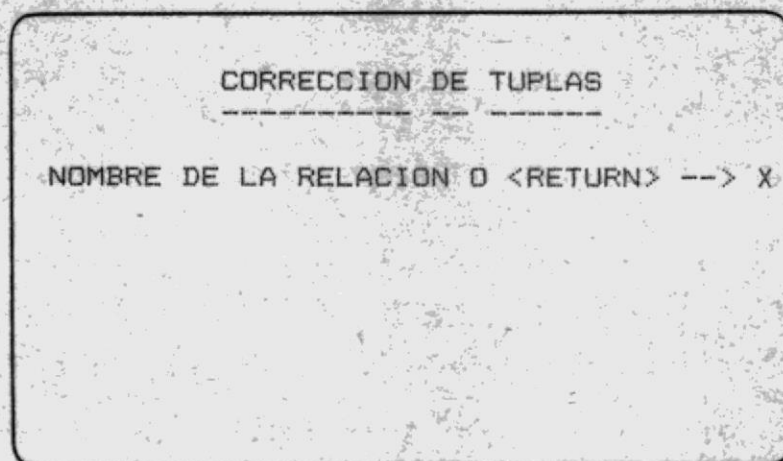
El usuario podrá eliminar las columnas presionando la tecla 'S' o cancelar el proceso digitando 'N'.

Una vez finalizada la eliminacion de las columnas marcadas, se podrá repetir dicho proceso si se presiona la tecla 'S'.

Si digita 'N' retorna al inicio de la opcion de "ELIMINACION DE DATOS".

5.4 CORRECCION DE TUPLAS

Esta opcion permite mantener las relaciones actualizadas, mediante la correccion de cualquier dato. Al ejecutar la opcion 4 de las presentadas en el Menu del EDITOR aparecera lo siguiente:



```

CORRECCION DE TUPLAS
-----
NOMBRE DE LA RELACION O <RETURN> --> X

```

Fig. 5.4.1

El usuario debera digitar el nombre de la relacion con la cual desea trabajar, o presionar <RETURN> si desea regresar al Menu del EDITOR.

Si los archivos para la relacion no existen, aparecera en la pantalla, el mensaje respectivo (Ver Fig. 5.4.2).

El usuario para continuar con el proceso debera presionar <RETURN> y digitar luego el nombre correcto de la relacion o insertar en el Drive #2 el diskette que contenga la relacion especificada.

```

CORRECCION DE TUPLAS
-----

NOMBRE DE LA RELACION O <RETURN> --> X

** NO EXISTEN ARCHIVOS PARA RELACION **
PRESIONE <RETURN> PARA CONTINUAR

```

Fig. 5.4.2

De esta misma forma se emitiran los mensajes correspondientes cuando la relacion no tiene tupla o columna.

Si los archivos de la relacion se encuentran disponibles, le seran presentados en pantalla el numero de tuplas de la relacion; nombres de las columnas y la estrategia de cada una de ellas.

```

CORRECCION DE TUPLAS
-----

NOMBRE DE LA RELACION O <RETURN> --> X
* ESTA RELACION CONSTA DE 14 TUPLAS *

COL.  NOMBRE  VALOR
----  -
1     NUMERO(E)
2     NOMBRE(S)
3     SUELDO(R)
4     ESTADO(C)

NUMERO DE TUPLA A CORREGIR -> _

```

Fig. 5.4.3

A continuacion, el usuario debera digitar el numero de la tupla que desea corregir y se le presentara en pantalla toda la informacion correspondiente a dicha tupla.

CORRECCION DE TUPLAS

NOMBRE DE LA RELACION O <RETURN> --> X
* ESTA RELACION CONSTA DE 14 TUPLAS *

COL.	NOMBRE	VALOR
1	NUMERO(E)	506
2	NOMBRE(S)	GISSELLE SALMON BOWEN
3	SUELDO(R)	99000.03
4	ESTADO(C)	S

NUMERO DE TUPLA A CORREGIR --> 2

DESEA CORREGIR TUPLA? S/N _

Fig. 5.4.4

Si el usuario desea corregir la tupla, debera seleccionar el numero que corresponda a la columna que desea modificar; no le sera aceptado un numero de columna diferente a los presentados.

CORRECCION DE TUPLAS		

NOMBRE DE LA RELACION O <RETURN> --> X		
* ESTA RELACION CONSTA DE 14 TUPLAS *		
COL.	NOMBRE	VALOR
----	-----	-----
1	NUMERO(E)	506
2	NOMBRE(S)	GISSELLE SALMON BOWEN
3	SUELDO(R)	99000.03
4	ESTADO(C)	S
NUMERO DE TUPLA A CORREGIR --> 2		
NUMERO DE COLUMNA A CORREGIR --> 3		

Fig. 5.4.5

Seguidamente, en la misma posición ocupada por el dato a modificar, el usuario deberá ingresar el dato correcto, el cual será validado dependiendo de la estrategia de la columna. Si es un entero, se aceptará un número positivo de hasta 5 cifras no mayor a 32767. Si es una cadena de caracteres, se aceptarán letras, números y blancos, siendo 50 el número máximo de caracteres. Si es un real, se aceptarán números y punto, pudiéndose digitar hasta 5 enteros y 2 decimales. Si es un carácter, solo se aceptará una letra o un número.


```

CORRECCION DE TUPLAS
-----

NOMBRE DE LA RELACION O <RETURN> --> X
* ESTA RELACION CONSTA DE 14 TUPLAS *

COL.      NOMBRE      VALOR
-----
1        NUMERO(E)    506
2        NOMBRE(S)    GISELLE SALMON BOWEN
3        SUELDO(R)    99600.00
4        ESTADO(C)    S

NUMERO DE TUPLA A CORREGIR  -> 2
NUMERO DE COLUMNA A CORREGIR -> 3

CORRIGE OTRA TUPLA? S/N _

```

Fig. 5.4.6

Si el usuario no desea corregir otra tupla, se le presentara la pantalla inicial de esta opcion; caso contrario debera seleccionar el numero de la siguiente tupla a corregir para continuar con el proceso.

5.5 CONSULTA DE TUPLAS

Este modulo del Editor de la Base de Datos Polirel. permite al usuario listar o desplegar cualquier relacion creada.

Presenta el siguiente menu de CONSULTA DE RELACIONES, donde el usuario debera elegir el medio a obtener los datos ya sea por consola o por impresora o retornar al MENU DEL EDITOR, presionando la tecla C(console) o P(printer) o S(salir) respectivamente. Luego debera digitar la relacion a consultar, si no existiera la relacion aparecera el mensaje respectivo; si no hubiere ninguna novedad se debera escoger la forma de presentacion de tuplas de la relacion

CONSULTA DE DATOS
=====

** C(CONSOLA) P(PRINTER) S(SAL) **

DIGITE RELACION --> A

DESEA 1-TABLA-TUPLA 2-UNA-TUPLA

Fig.5.5.1

CASO 1: TABLA-TUPLA:

Se presentara el numero de tuplas que posee la relacion y las columnas de la relacion, el usuario podra seleccionar la(s) columna(s) que no desea ver marcando con una X al nivel de la columna a no visualizar. Cabe senalar que para el caso de una columna de estrategia de tipo string a visualizar, se asumira un despliegue de hasta diez caracteres del dato; el usuario podra cambiar el numero de caracteres a desplegar (desde 1 a 20) digitando dicha cantidad al nivel de la columna a visualizar, previamente presionando la tecla de Barra Espaciadora.

NOTA:

El usuario debera considerar que solo podra desplegar un maximo de 80 caracteres por tupla; para ello debera seleccionar en la forma mas conveniente las columnas a consultar. El numero de caracteres por tupla a visualizar se formara por:

- 2 caracteres para el marco
- 2 caracteres por cada columna estrategia tipo caracter.
- 6 caracteres por cada columna estrategia tipo entero.

9 caracteres por cada columna estrategia tipo real.

10+1 caracteres por cada columna estrategia tipo string(assume 10).

PROCESO DE LA RELACION A
EXISTEN 20 TUPLAS

NRO EST COLUM

1-(E)-CODIGO

2-(S)-NOMBRE 15

3-(R)-SUELDO X

* MARQUE CON UNA "X" LAS *
* COLUMNAS QUE NO DESEA VER *
* CON BLANCO LAS DESEADAS *
* DIGITE "Q" PARA SALIR *

Fig.5.5.2

Una vez seleccionado, presionando la tecla "Q" se desplegaran las tuplas en forma horizontal en numero de hasta diez por pantalla o de cincuenta por papel, se mostraran los nombres de las columnas referenciados por el numero de columna que le corresponde.

PROCESO DE LA RELACION A	
NRO.	TUPLA
1	
*1	2
* 1001	AGUSTIN CHUSAN
*11001	FERNANDO PAEZ
*92399	CAROLINA PEREZ
*	
*	
*	
*	
*	
*	
*	
*1-CODIGO	
*2-NOMBRE	

Fig.5.5.3

CASO 2: UNA TUPLA

Se desplegara el numero de tuplas que posee la relacion y se visualizara la primera tupla de la relacion en forma vertical.

Solo se desplegara una tupla a la vez por pantalla o impresora, con todas sus columnas y sus datos completos.

```
PROCESO DE LA RELACION A  
EXISTEN 20 TUPLAS  
NRO. TUPLA 1  
  
NRO EST COLUM  
1-(E)-CODIGO 1001  
2-(S)-NOMBRE ABUSTIN CHUSAN VITERI  
3-(R)-SUELDO 12000.00
```

Fig.5.5.4

SELECCION DE TUPLAS:

Una vez desplegada la(s) tupla(s) segun CASO 1 o CASO 2, el usuario podra hacer uso de los siguientes comandos que aparecieran en la parte superior de la pantalla.

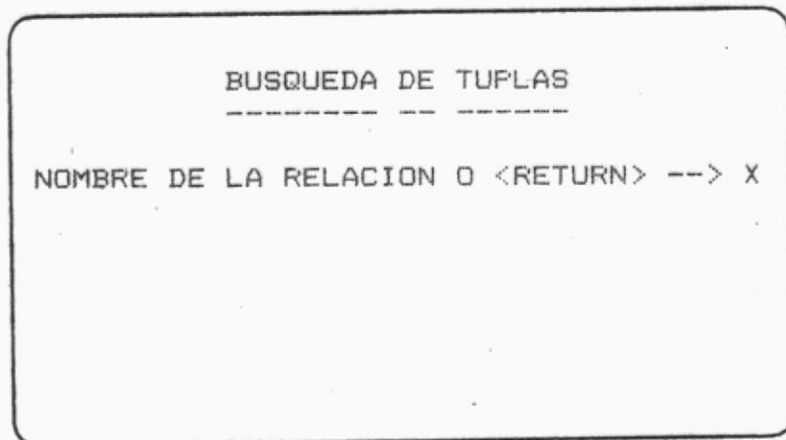
P(PRO R(RELAT A(ATR I(INI S(SAL _

Donde:

- P : permite ver la proxima tupla.
- R : permite ver una tupla especifica para ello debera digitar el numero de tupla que se desea ver.
- A : permite ver la tupla anterior.
- I : permite ver la primera tupla.
- S : permite retornar al Menu de "CONSULTA DE DATOS".

5.6 BUSQUEDA DE TUPLAS

Esta opcion permite presentar por pantalla, todas aquellas tuplas que satisfagan un criterio de busqueda. Al ejecutar la opcion 6 de las presentadas en el Menu del EDITOR aparecera lo siguiente:



```
BUSQUEDA DE TUPLAS
-----
NOMBRE DE LA RELACION O <RETURN> --> X
```

Fig. 5.6.1

El usuario debera digitar el nombre de la relacion con la que desea trabajar, o presionar <RETURN> si desea regresar al Menu del EDITOR.

Si los archivos no existen, aparecera en la pantalla, el mensaje respectivo. El usuario para continuar con el proceso debera presionar <RETURN> y digitar luego el nombre correcto de la relacion o insertar en el Drive #2 el diskette que contenga la relacion especificada.


```

      BUSQUEDA DE TUPLAS
      -----

NOMBRE DE LA RELACION O <RETURN> --> X

** NO EXISTEN ARCHIVOS PARA RELACION **
PRESIONE <RETURN> PARA CONTINUAR

```

Fig. 5.6.2

De esta misma forma se emitiran los mensajes correspondientes cuando la relacion no tiene tupla o columna.

Si los archivos de la relacion se encuentran disponibles, le seran presentados en pantalla los nombres de las columnas y la estrategia de cada una de ellas.

```

      BUSQUEDA DE TUPLAS
      -----

NOMBRE DE LA RELACION O <RETURN> --> X

COL.  NOMBRE  VALOR
-----
  1  NUMERO(E)
  2  NOMBRE(S)
  3  SUELDO(R)
  4  ESTADO(C)

NUMERO DE COLUMNA --> -
VALOR A BUSCAR   --> -

```

Fig. 5.6.3

Seguidamente, el usuario debera seleccionar el numero de la columna sobre la cual se va a efectuar la busqueda; no le sera aceptado un numero diferente a los presentados.

El ingreso del dato correspondiente al Valor a Buscar, sera validado dependiendo de la estrategia de la columna. Si es un entero, se aceptara un numero positivo de hasta 5 cifras no mayor a 32767. Si es una cadena de caracteres, se aceptaran letras, numeros y blancos; siendo 20 el numero maximo de caracteres. Si es un real, se aceptaran numeros y punto, pudiendose digitar hasta 5 enteros y 2 decimales. Si es un caracter, solo se aceptara una letra o un numero.

BUSQUEDA DE TUPLAS		

NOMBRE DE LA RELACION O <RETURN> --> X		
COL.	NOMBRE	VALOR
----	-----	-----
1	NUMERO(E)	
2	NOMBRE(S)	
3	SUELDO(R)	
4	ESTADO(C)	
NUMERO DE COLUMNA -->		3
VALOR A BUSCAR -->		4006.1

Fig. 5.6.4

A continuacion, se inicia el proceso de busqueda. Si la estrategia de la columna seleccionada es una cadena de caracteres, la busqueda se realiza hasta encontrar la

primera ocurrencia dentro de la cadena de caracteres de la tupla corriente.

Si el dato a buscar no existe, un mensaje le sera presentado de la siguiente forma:

** VALOR NO ENCONTRADO **
PRESIONE <RETURN> PARA CONTINUAR

el usuario debera presionar <RETURN> para regresar al inicio de esta opcion.

Si la busqueda fue satisfactoria, se incrementa el contador de tuplas encontradas y le sera presentada en pantalla toda la informacion correspondiente a la tupla que tenga dicho valor.

BUSQUEDA DE TUPLAS		
NOMBRE DE LA RELACION COL. NOMBRE		O <RETURN> --> X VALOR
1 NUMERO(E)		8405
2 NOMBRE(2)		GISSELLE SALMON BOWEN
3 SUELDO(R)		4006.10
4 ESTADO(C)		S
NUMERO DE COLUMNA --> 3		
VALOR A BUSCAR --> 4006.10		
TUPLA CORRIENTE --> 3		
TUPLAS ENCONTRADAS--> 1		
OTRA TUPLA? S/N _		

Fig. 5.6.5

Si el usuario desea que se le presente en pantalla la siguiente tupla, continuara el proceso de busqueda a partir de la ultima tupla presentada; caso contrario se le presentara la pantalla inicial de esta opcion.

6. QUERY

Constituye el Procesador de Consulta para la Base: se presentara de la siguiente manera:

A screenshot of a terminal window showing the POLIREL 2.0 interface. The text is as follows:

```
BASE DE DATOS
-----
POLIREL 2.0
-----

USER-ID (7-DIG.) -> _
```

Fig. 6.1

El usuario debera digitar un numero que lo identifique, pudiendo ser este el numero de la matricula, que servira para llevar un control sobre la sesion de trabajo dentro del Query.

Luego se presentara una segunda pantalla para seleccionar el modo de proceso para las expresiones de Consulta (Ver Fig. 6.2).

```

** BASE DE DATOS RELACIONAL **

QUERY
=====

DESEA PROCESAR
-----

1.- UNA EXPRESION
2.- UNA CONSULTA CATALOGADA
3.- RETORNA AL MENU PRINCIPAL

DIGITE OPCION --->

LIBRE:SELEC 25  JOIN 25  PROY 25  T GRAL 50

```

Fig. 6.2

Al escoger la opcion 1 aparecera una pantalla que se resume en la figura 6.3.

```

**  BASE DE DATOS RELACIONAL  **

      QUERY
      =====

DESEA PROCESAR UNA EXPRESION (S/N) -> S

DIGITE EXPRESION

.....

LIBRE:SELEC 25  JOIN 25  PROY 25  T GRAL 50

```

Fig. 6.3

El usuario podra cancelar su opcion y retornar al Menu anterior digitando N.

El Query lee un enunciado (expresion), lo valida y finalmente lo interpreta para que sea ejecutado por los diferentes procesos.

Los enunciados deben tener una longitud maxima de 70 caracteres, pudiendo contener todos los espacios que sean necesarios para su mayor legibilidad.

6.1 RESULTADOS TEMPORALES Y PERMANENTES

6.1.1 TEMPORALES

Toda operacion (Join, Seleccion, Proyeccion) que no contemple un enunciado de asignamiento sera considerado temporal, es decir, toda operacion que deba ser realizada como paso intermedio para la obtencion de un resultado final, y por este hecho no podra ser utilizada por el usuario en enunciados sucesivos.

El sistema asignara nombres a todas aquellas estructuras temporales generadas por operaciones del mismo tipo.

Por ejemplo:

JA := A : CODIGO > D * B

En este ejemplo el sistema creara una estructura temporal para la seleccion y asi poder realizar el proceso de join con la relacion B.

6.1.2 PERMANENTES

Toda estructura y operacion que sea especificada mediante un enunciado de asignamiento sera considerada PERMANENTE por el sistema.

Toda estructura permanente, por consiguiente, podra ser utilizada en enunciados posteriores,

siempre y cuando sea permitido su uso segun las reglas sintacticas establecidas mas adelante.

Del ejemplo anterior JA es considerada por el sistema como una estructura permanente.

6.2 CONSULTAS CATALOGADAS

Cuando se escoge la opción de procesar una consulta catalogada, se presenta la siguiente pantalla:

```

** BASE DE DATOS RELACIONAL **

      QUERY
      =====

DESEA PROCESAR UNA CONSULTA S/N-> S
DIGITE EL NOMBRE DE LA CONSULTA-> A

1>PA:=SB%NOMBRE

LIBRE:SELEC 25  JOIN 25  PROY 25  T GRAL 50
```

Fig. 6.2.1

Si el usuario no desea procesar una consulta, retorna al menú del QUERY (Ver Fig. 6.2). Caso contrario debe digitar el nombre de la consulta y aparecerán en la pantalla una a una las expresiones que se van procesando. El usuario deberá presionar <RETURN> después de cada expresión que sea presentada.

6.3 OPERADOR SELECCION

Mediante este operador se realiza el seleccionamiento de tuplas dentro de una relacion, que satisfacen una condicion dada.

La condicion esta compuesta por OPERANDOS, OPERADORES, CONSTANTES y puede ser simple o compuesta.

- operandos: Son los nombres de columnas de una relacion.
- OPERADORES:
 - Relacionales: >, <, =
 - Logicos: AND, OR
- CONDICION SIMPLE: No tiene operadores logicos.
- CONDICION COMPUESTA: Se forma por 2 condiciones simples y un operador logico.

Por ejemplo, si consideramos la siguiente relacion A:

CUENTA	NOMBRE	ESTADO	SUELDO
1	JUAN PEREZ	C	20000.00
2	VICTOR LEON	C	30000.00
3	ALFONSO VARGAS	S	50000.00
4	JOSE RUMBEA	D	70000.00
5	JAVIER PAZ	V	40000.00

Una condicion simple seria:

SA := A : ESTADO = C

Donde SA va a pasar a ser una estructura de tipo permanente, formada por las tuplas 1 y 2 que son las que satisfacen la condicion.

Una condicion compuesta podria ser:

SB := A : NOMBRE = /ALFONSO VARGAS/ OR SUELDO > 60000.00

En este caso la estructura estara formado por las tuplas 3 y 4.

6.4 OPERADOR JOIN

El proceso JOIN nos da como resultado una nueva estructura a partir de dos relaciones dadas.

Unicamente se puede realizar el JOIN cuando existe solamente una columna en comun para ambas relaciones, estas deben tener el mismo nombre y ser del mismo tipo (estrategia).

El operador JOIN se puede ejecutar sobre:

- 1) Dos Relaciones
- 2) Una Relacion y una Seleccion

Tanto para el primero como para el segundo caso los resultados son identicos.

Por ejemplo, sean las Relaciones:

A			B	
NUMERO	TIPO	COLOR	TIPO	COSTO
1	PLUMA	AZUL	PLUMA	20.00
2	LAPIZ	ROJO	LAPIZ	5.00
3	REGLA	LILA	REGLA	10.00
4	PLUMA	NEGRO		

Si ejecutamos el siguiente enunciado:

JA := A * B

Tendriamos una estructura compuesta por las siguientes tuplas:

ESTRUCTURA JAB

NUMERO	TIPO	COLOR	COSTO
1	PLUMA	AZUL	20.00
2	LAPIZ	ROJO	5.00
3	REGLA	LILA	10.00
4	PLUMA	NEGRO	20.00

Si ejecutamos, JB := B * A tendremos:

ESTRUCTURA JBA

TIPO	COSTO	NUMERO	COLOR
PLUMA	20.00	1	AZUL
PLUMA	20.00	4	NEGRO
LAPIZ	5.00	2	ROJO
REGLA	10.00	3	LILA

6.5 OPERADOR PROYECCION

Mediante este operador se selecciona columnas específicas de una relacion.

El operador proyeccion se puede ejecutar sobre:

- a) Una Relacion
- b) Una Seleccion
- c) Un Join

Para poder realizar una proyeccion se debe especificar las columnas que se desea proyectar.

Si tenemos la Relacion A:

NUMERO	TIPO	CODIGO	COLOR	VALOR
1	PLUMA	XP	ROJO	20.00
2	LAPIZ	XL	AZUL	5.00
3	REGLA	XR	VERDE	15.00

y se quiere proyectar las columnas TIPO y VALOR tendremos:

PA := A % TIPO, VALOR

PROYECCION PA

TIPO	VALOR
PLUMA	20.00
LAPIZ	5.00
REGLA	10.00

Para los casos b) y c) las estructuras resultantes son iguales.

Basta que una de las columnas a proyectarse no exista para que no se pueda realizar la proyeccion, dando el mensaje de error correspondiente.

6.6 COMANDO LIST / DISP

Este comando del Query permite al usuario listar o desplegar cualquier relacion, seleccion, join o proyeccion que sean estructuras permanentes.

Para el caso de join o proyeccion las tuplas se presentaran en forma vertical (una tupla a la vez).

Para el caso de relacion o seleccion se podra escoger la forma de presentacion de las tuplas:

1.- tabla de tuplas

2.- por una tupla

1. Tabla de tuplas:

Presenta los datos de una tupla en forma horizontal, tupla por tupla, maximo hasta diez tuplas si es por consola, y hasta 50 tuplas por pagina si es por impresora. Tambien le permite al usuario seleccionar las columnas a visualizar.

2. Por una tupla:

Presenta los datos de una tupla en forma vertical, cada columna en una linea.



```

CONSULTA DE DATOS
===== ** =====

** C(CONSOLA) P(PRINTER) S(SAL) **

DIGITE RELACION --> A

*DESEA 1-TABLA-TUPLA 2-UNA-TUPLA*

```

Fig.6.6.1

CASO 1: TABLA-TUPLA:

Se presentara el numero de tuplas que posee la relacion y las columnas de la relacion, de las cuales el usuario podra seleccionar aquellas que no desea ver marcando con una X al nivel de la columna a no visualizar. Cabe senalar que para el caso de una columna de estrategia string a visualizar, se asumira un despliegue de hasta diez caracteres del dato; el usuario podra cambiar el numero de caracteres a desplegar (desde 1 a 20) digitando dicha cantidad al nivel de la columna a visualizar, previamente presionando la tecla de Barra Espaciadora.

NOTA:

El usuario debera considerar que solo podra desplegar un maximo de 80 caracteres por tupla; para ello debera seleccionar en la forma mas conveniente las columnas a visualizar. El numero de caracteres por tupla a visualizar se formara por:

- 2 caracteres para el marco
- 2 caracteres por cada columna estrategia tipo caracter.
- 6 caracteres por cada columna estrategia tipo entero.
- 9 caracteres por cada columna estrategia tipo real.
- 10+1 caracteres por cada columna estrategia tipo string(assume 10).

PROCESO DE LA RELACION A
EXISTEN 20 TUPLAS

NRO EST COLUM

1-(E)-CODIGO

2-(S)-NOMBRE 15

3-(R)-SUELDO X

* MARQUE CON UNA "X" LAS *
* COLUMNAS QUE NO DESEA VER *
* CON BLANCO LAS DESEADAS *
* DIGITE "Q" PARA SALIR *

Fig.6.6.2

Una vez seleccionado, presionando la tecla "Q" se desplegaran las tuplas en forma horizontal en numero de hasta diez por pantalla o de cincuenta por papel, se mostraran los nombres de las columnas referenciados por el numero de columna que le corresponde.

PROCESO DE LA RELACION A

NRO. TUPLA 1

*1	2
* 1001	AGUSTIN CHUSAN VITERI
*90901	PILAR DE CABRERA
*11001	ANGEL ZAMORA
*92399	CAROLINA PEREZ
*	
*	
*	
*	
*	
*	
*	
*1-CODIGO	
*2-NOMBRE	

Fig.6.6.3

CASO 2: UNA TUPLA

Se desplegara el numero de tuplas que posee la relacion y se visualizara la primera tupla de la relacion en forma vertical.

Solo se presentara una tupla a la vez por pantalla o impresora, con todas sus columnas y sus datos completos.

PROCESO DE LA RELACION A	
EXISTEN 20 TUPLAS	
NRO. TUPLA	1
NRO EST COLUM	
1-(E)-CODIGO	1001
2-(S)-NOMBRE	AGUSTIN CHUSAN VITERI
3-(R)-SUELDO	12000.00

Fig.6.6.4

SELECCION DE TUPLAS:

Una vez desplegada la(s) tupla(s) segun CASO 1 o CASO 2, el usuario podra hacer uso de los siguientes comandos que apareceran en la parte superior de la pantalla.

```

P (PRO R (RELAT A (ATR I (INI S (SAL - - -
      PROCESO DE LA RELACION A
      EXISTEN 20 TUPLAS
      NRO. TUPLA 1

      NRO EST COLUM
      1-(E)-CODIGO 1001
      2-(S)-NOMBRE AGUSTIN CHUSAN VITERI
      3-(R)-SUELDO 12000.00

```

Fig.6.6.5

Donde:

Para la visualizacion de tuplas existe el siguiente menu:

P (proximo	permite ver la siguiente tupla.
A (atras	permite ver la tupla anterior..
I (inicio	permite ver la tupla inicial de la relacion.
R (relativo	dado un numero relativo de tupla logica, permite ver dicha tupla.
S (salir	retorna a pedir una nueva expresion.

7. UTILITARIAS

7.1 FORMATEO DE ARCHIVOS

Cada relacion en su organizacion fisica esta formada por tres archivos, dos de los cuales, el archivo de tuplas (AT) y el de datos alfanumericos (ADA) almacenan los datos para esa relacion. El archivo descriptor de tuplas (ADT) que es el tercero, contiene informacion general para la relacion. La opcion FORMATEO DE ARCHIVOS permite al usuario definir lo siguiente:

- . El nombre de la relacion
- . Tamano en bloques para archivo AT
- . Tamano en bloques para archivo ADA.

Al seleccionar esta opcion del menu UTILITARIAS, se le presenta al usuario una primera pantalla (Ver Fig. 7.1.1), donde debe ingresar un caracter alfabetico que identifique la relacion a formatearse, o presionar la tecla <RETURN> si desea retornar al menu UTILITARIAS.

FORMATEO DE ARCHIVOS

. NOMBRE DE RELACION O <RETURN> -> _

Fig. 7.1.1

Los siguientes parametros a ingresarse corresponden al

tamaño de los archivos AT y ADA, para lo cual el usuario debe guiarse por lo siguiente:

- Tanto el archivo AT y ADA pueden tener máximo diez bloques.
- Cada bloque puede almacenar 512 caracteres.
- La longitud de la tupla depende de las estrategias de las columnas y la longitud asociada a cada una es:

ESTRATEGIA	LONGITUD
C	1 carac.
E	5 carac.
R	7 carac.
S	7 carac.

Su valor se calcula de la siguiente forma:

$$LT = 1 + LE(1) + LE(2) + \dots + LE(N)$$

LT = Longitud de tupla

1 = para caracter de anulacion de tupla

LE = Longitud de la estrategia

N = Numero de columnas de la relacion.

Si la relacion tiene cuatro columnas con diferente estrategia entonces:

$$LT = 1 + 1 + 5 + 7 + 7 = 21 \text{ caract.}$$

- El numero de bloques para el archivo AT se calcula aproximadamente así:

$$NB = (NT * LT) / 512 + 1$$

NB = Numero de bloques para archivo AT

NT = Numero de tuplas de la relacion (valor estimado)

LT = Longitud de la tupla

Si la relacion va a contener 10 tuplas entonces:

$$NB = (10 * 21) / 512 + 1 = 1.4$$

NB = maximo 2 bloques

El numero de bloques para el archivo ADA depende de cuantas columnas con estrategia tipo string(S) y cuantas tuplas (valor estimado) tiene la relacion.

FORMATEO DE ARCHIVOS

NOMBRE DE RELACION O <RETURN> -> A

BLOQUES (1..10)

PARA ARCHIVO AT -> 2

PARA ARCHIVO ADA -> 2

Fig. 7.1.2

Luego de ingresar el numero de bloques para cada Archivo aparecera un mensaje para indicar que se esta

formateando los archivos (Ver Fig. 7.1.3)

```

      FORMATEO DE ARCHIVOS
      -----

NOMBRE DE RELACION O <RETURN> -> A

      # BLOQUES (1..10)

PARA ARCHIVO AT -> 2
PARA ARCHIVO ADA -> 2

FORMATEANDO.....

```

Fig. 7.1.3

Una vez finalizado el proceso se regresara automaticamente al Menu de las UTILITARIAS.



7.2 INDICES INVERTIDOS

Esta opcion permite generar un indice del archivo de tuplas (AT) de una relacion, en virtud de los diferentes valores que pueda tener uno de sus atributos.

Este indice hara posible que el usuario pueda acceder a determinadas tuplas de una relacion directamente, bajo cierto criterio de seleccion. La utilizacion del Indice es automatica a traves del proceso de Seleccion dentro del Query (siendo totalmente transparente al usuario), del Generador de Reportes o de cualquier programa PASCAL que interactue con la Base utilizando la Unidad Polirel.

Al ejecutar la opcion 2 de las presentadas en el Menu de UTILITARIAS aparecera lo siguiente:

* GENERACION DE ARCHIVOS *

* INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> X

Fig.7.2.1

El usuario debera digitar el nombre de la relacion con

la que desea trabajar.

Seguidamente, si los archivos de la relacion se encuentran disponibles, le seran presentados en pantalla todos los nombres de las columnas numeradas, con su respectiva estrategia. (Ver Fig. 7.2.2)

```

* GENERACION DE ARCHIVOS *
* INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> A

ESTA RELACION CUENTA CON 99 TUPLAS Y
LAS SIGUIENTES COLUMNAS:

    1 NUMERO (E)          2 NOMBRE (S)
    3 SUELDO (R)          4 ESTADO (C)

NUMERO DE COLUMNA SOBRE LA QUE SE
VA A GENERAR EL INDICE --> _
  
```

Fig. 7.2.2

```

* GENERACION DE ARCHIVOS *
* INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> B

* ARCHIVOS DE LA RELACION NO EXISTEN *
  
```

Fig. 7.2.3



BIBLIOTECA

Si los archivos no existen, aparecera en la parte inferior de la pantalla, el mensaje respectivo. (Ver fig. 7.2.3) El Usuario para continuar con el proceso debera presionar <RETURN> y digitar luego el nombre correcto de la relacion o insertar en el DRIVE #2 el diskette que contenga la relacion especificada, segun sea el caso.

El usuario entonces debera seleccionar el numero que corresponda a la columna sobre la que se desea generar el Indice. Si el numero digitado no corresponde a alguno de los presentados, este no sera aceptado, emitiendo el correspondiente error (Ver Fig. 7.2.4); dandole al usuario la oportunidad de corregir el dato y continuar el proceso al presionar la tecla <RETURN>.

```

      * GENERACION DE ARCHIVOS *
      * INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> A

ESTA RELACION CUENTA CON 99 TUPLAS Y
LAS SIGUIENTES COLUMNAS:

      1 NUMERO (E)           2 NOMBRE (S)
      3 SUELDO (R)          4 ESTADO (C)

NUMERO DE COLUMNA SOBRE LA QUE SE
VA A GENERAR EL INDICE --> 5

      * NUMERO DE COLUMNA ERRADO *
```

Fig. 7.2.4

Luego, para efectos de poder calcular el número de bloques que deberá tener el archivo de Índices, el usuario deberá digitar el número aproximado de valores diferentes que existen para ese atributo dentro de la relación, teniendo en cuenta el máximo señalado, el mismo que dependerá de la estrategia de dicho atributo. (Refierase al Manual del Sistema, Numeral 8.2)

```

* GENERACION DE ARCHIVOS *
* INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> A

ESTA RELACION CUENTA CON 99 TUPLAS Y
LAS SIGUIENTES COLUMNAS:

      1 NUMERO (E)           2 NOMBRE (S)
      3 SUELDO (R)          4 ESTADO (C)

NUMERO DE COLUMNA SOBRE LA QUE SE
VA A GENERAR EL INDICE --> 4

NUMERO DE VALORES DIFERENTES DENTRO
DE LA COLUMNA (MAXIMO: 99) --> 100

* NUMERO MAXIMO ERRADO *

```

Fig. 7.2.5

Si el número digitado excede el límite (Ver Fig. 7.2.5), un mensaje de error aparecerá, pudiendo el usuario, al presionar <RETURN>, corregir el número y continuar el

proceso; el mismo que dara la oportunidad de que el usuario cancele el proceso o lo continúe:

```
* GENERACION DE ARCHIVOS *
* INDICES INVERTIDOS *

NOMBRE DE LA RELACION --> A

ESTA RELACION CUENTA CON 99 TUPLAS Y
LAS SIGUIENTES COLUMNAS:

      1 NUMERO (E)           2 NOMBRE (S)
      3 SUELDO (R)          4 ESTADO (C)

NUMERO DE COLUMNA SOBRE LA QUE SE
VA A GENERAR EL INDICE --> 4

NUMERO DE VALORES DIFERENTES DENTRO
DE LA COLUMNA (MAXIMO: 99) --> 5

DIGITE <RETURN> PARA CONTINUAR
      N <RETURN> PARA RETORNAR AL MENU
```

Fig. 7.2.6

Si se cancela, se regresara al Menu de las UTILITARIAS sin que nada haya sido generado ni modificado; si el usuario continua con el proceso la siguiente pantalla le sera mostrada:

```
GENERANDO INDICE ...
```

Fig. 7.2.7

la misma que indicara, con el movimiento del cursor, que el proceso esta avanzando. Una vez finalizado este, se regresara automaticamente al Menu de las UTILITARIAS, habiendo sido creado en el diskette que se encuentre en el DRIVE #2 el archivo de Indices Invertidos; a menos que cualquier error fatal se haya producido durante el proceso, el mismo que se hara notar al usuario, debidamente.

Importante:

El usuario es el unico responsable de actualizar el Indice, generandolo nuevamente, cada vez que la relacion es modificada ya sea adicionando, eliminando o corrigiendo tuplas si es el caso. De no hacerlo, los resultados obtenidos por medio del Indice podrian no ser correctos.

7.3 COLECCION DE BASURA

Al momento en que se empieza a ejecutar la opción de colección de basura, aparece la pantalla de la siguiente forma:

```
*****
*                                     *
*      COLECCION DE BASURA          *
*                                     *
*****

DIGITE EL NOMBRE DE LA RELACION
O PRESIONE <RETURN>-----> A
```

Fig.7.3.1

Se deberá digitar la letra que identifica la relación que quiere coleccionar, o presionar <RETURN> si desea regresar al menú de las UTILITARIAS.

El nombre de la relación será aceptado solo si cumple los siguientes requerimientos:

- Deben existir los archivos físicos.
- La relación debe tener por lo menos una tupla y una columna.

Una vez que la relación ha sido aceptada, se realiza el

proceso y la pantalla estara asi:

```

*****
*                               *
*   .  COLECCION DE BASURA   *
*                               *
*****

DIGITE EL NOMBRE DE LA RELACION
O PRESIONE <RETURN>-----> A

COLECCIONANDO.....

```

Fig.7.3.2

Cuando termina el proceso aparecera en la parte inferior de la pantalla un mensaje:

FIN DE COLECCION DE BASURA.....

Una vez finalizado el proceso se regresara automaticamente al Menu de las UTILITARIAS.

7.4 CLASIFICACION DE RELACIONES

Esta opcion permite generar un Archivo de Clasificacion ordenando en forma ascendente una relacion dada, en virtud de una llave primaria (numero de columna) y si se requiere de una llave secundaria.

Este archivo queda a disposicion del Generador de Reportes y de cualquier usuario de la Base de Datos, que utilice la Unidad Polirel.

Al ejecutar la opcion 4 de las presentadas en el Menu de UTILITARIAS aparecera lo siguiente:

CLASIFICACION DE RELACIONES

NOMBRE DE LA RELACION O <RETURN> -> X

Fig. 7.4.1

El usuario debera digitar el nombre de la relacion con la cual desea trabajar, o presionar <RETURN> si desea regresar al menu de las UTILITARIAS.

Si los archivos no existen, aparecera en la parte inferior de la pantalla, el mensaje respectivo. El usuario para continuar con el proceso debera presionar

<RETURN> y digitar luego el nombre correcto de la relacion o insertar en el Drive #2 el diskette que contenga la relacion especificada.

CLASIFICACION DE RELACIONES

NOMBRE DE LA RELACION O <RETURN> -> X

** NO EXISTEN ARCHIVOS PARA RELACION **
PRESIONE <RETURN> PARA CONTINUAR

Fig. 7.4.2

De esta misma forma, se emitiran los mensajes correspondientes cuando la relacion no tiene tupla o columna.

Si los archivos de la relacion se encuentran disponibles, le seran presentados en pantalla todos los nombres de las columnas y sus estrategias.

```

CLASIFICACION DE RELACIONES
-----
NOMBRE DE LA RELACION  O <RETURN> -> X

      COLUMNA      NOMBRE
      -----
      1            NUMERO (E)
      2            NOMBRE (S)
      3            SUELDO (R)
      4            ESTADO (C)

```

Fig. 7.4.3

A continuacion, si la relacion tiene mas de una columna
le sera presentada la siguiente pantalla:

```

CLASIFICACION DE RELACIONES
-----
NOMBRE DE LA RELACION  O <RETURN> -> X

      COLUMNA      NOMBRE
      -----
      1            NUMERO (E)
      2            NOMBRE (S)
      3            SUELDO (R)
      4            ESTADO (C)

DIGITE EL NUMERO DE LA COLUMNA
PARA LLAVE PRIMARIA  --> -
PARA LLAVE SECUNDARIA--> -

```

Fig. 7.4.4

El usuario ,entonces debera seleccionar el numero que corresponda a la columna que desea considerar como llave primaria (columna de mas alto nivel); no le sera aceptado un numero diferente a los presentados.

Para llave secundaria (columna de nivel inferior), debera digitar el numero correspondiente o presionar <RETURN> si no requiere dicha llave.

Si la columna primaria es una cadena de caracteres, el proceso de clasificacion solo considera los 20 primeros caracteres.

Una vez ingresado los parametros anteriores, el proceso de clasificacion esta listo para ejecutarse, presentando la siguiente pantalla:

CLASIFICACION DE RELACIONES

NOMBRE DE LA RELACION O <RETURN> --> X

COLUMNA -----	NOMBRE -----
1	NUMERO(E)
2	NOMBRE(S)
3	SUELDO(R)
4	ESTADO(C)

DIGITE EL NUMERO DE LA COLUMNA
PARA LLAVE PRIMARIA --> 3
PARA LLAVE SECUNDARIA--> 2

CLASIFICANDO...

Fig. 7.4.5

la misma que indicara con el movimiento del cursor, que el proceso se esta efectuando. Al finalizar este, un mensaje le sera presentado para presionar <RETURN> y regresar al Menu de las UTILITARIAS; habiendo sido creado en el diskette que se encuentra en el drive #2, el Archivo de Clasificacion.

Se puede generar un Archivo de Clasificacion por cada columna de la relacion.

Importante:

El usuario es el unico responsable de actualizar el archivo de clasificacion, generandolo nuevamente cuando la relacion es modificada.

7.5 DESCARGA DE RELACIONES

Esta opcion permite al usuario generar un archivo para el mantenimiento de la relacion. La opcion DESCARGA DE RELACIONES debe ser ejecutada en los siguientes casos:

- . Si la relacion tiene demasiadas tuplas eliminadas
- . Si no hubo suficiente espacio en los archivos (AT y/o ADA), durante el Ingreso de Tuplas.

Luego de ingresar el nombre de la relacion, si ya existe el archivo de reorganizacion generado, aparece una opcion para eliminarlo o no (Ver Fig. 7.5.1)

DESCARGA

NOMBRE DE RELACION O <RETURN> -> A

YA EXISTE ARCHIVO DE REORGANIZACION
DESEA ELIMINARLO? (S/N) -> _

Fig. 7.5.1

A continuacion se presenta un mensaje que indica que el proceso se esta ejecutando.

DESCARGA

NOMBRE DE RELACION O <RETURN> -> A

PROCESANDO.....

Fig. 7.5.2

7.6 CARGA DE RELACIONES

Esta opcion de las UTILITARIAS utiliza el archivo de reorganizacion, creado en forma transparente para el usuario, por la opcion Descarga de Relaciones.

Con este archivo genera nuevamente los archivos (ADT, AT Y ADA) para la relacion.

Al seleccionar esta opcion una pantalla inicial (Ver Fig. 7.6.1) sera presentada al usuario, donde digitara el nombre de la relacion, sin presionar <RETURN>. Los mensajes de error apareceran en la parte inferior de la pantalla

CARGA

NOMBRE DE RELACION D <RETURN> -> A

** NO EXISTE ARCHIVO DE REORGANIZACION **
PRESIONE <RETURN> PARA CONTINUAR



Fig. 7.6.1

En caso de no existir error, en la parte inferior de la pantalla aparecera un mensaje para indicar que el

proceso de CARGA DE RELACIONES se esta ejecutando.

CARGA

NOMBRE DE RELACION O <RETURN> -> A

PROCESANDO.....

Fig. 7.6.3

Importante:

Cuando los nuevos archivos AT y ADA han sido formateados con un menor numero de bloques al que tenian originalmente, los nuevos archivos almacenaran informacion incompleta, emitiendose uno de estos mensajes:

- * NO HAY ESPACIO EN ARCHIVO AT *
- * NO HAY ESPACIO EN ARCHIVO ADA *

7.7 CONSULTAS CATALOGADAS

El modulo de Consultas Catalogadas permite al usuario crear, consultar, modificar y borrar sus consultas a traves de menus que lo van guiando en forma clara sencilla.

El nombre de todas las consultas debe cumplir ciertas condiciones:

- 1.- No debe tener mas de dos caracteres.
- 2.- El primer caracter debe ser una letra.

Siempre que el nombre que se digite no cumple estas condiciones aparece el mensaje:

RESPUESTA ERRADA, REINTENTE

Cuando se comienza a ejecutar la opcion de consultas catalogada aparece el menu principal:

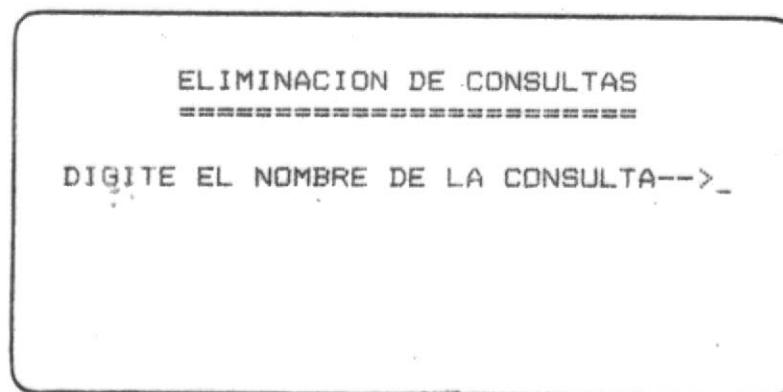
```
*****
*                                     *
*      CONSULTAS CATALOGADAS      *
*                                     *
*****

1.- ELIMINACION
2.- CREACION
3.- MODIFICACION
4.- CONSULTA
5.- FIN

DIGITE LA OPCION DESEADA--
```

Fig. 7.7.1

- Si el usuario digita '1', aparece en la pantalla:



ELIMINACION DE CONSULTAS
=====

DIGITE EL NOMBRE DE LA CONSULTA-->_

Fig.7.7.2

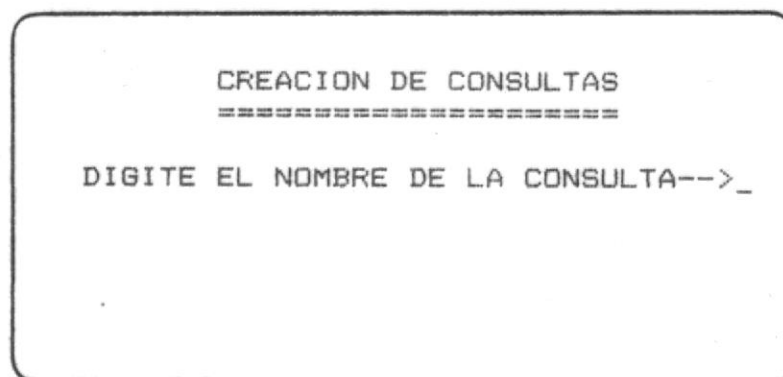
Si el nombre de la consulta que se da es correcto pero el archivo no existe, aparece el mensaje:

ARCHIVO NO EXISTE

Para regresar al menu de las Consultas Catalogadas se debe presionar <RETURN>.

Si todo esta correcto y por lo tanto no hay mensajes de error, la consulta es eliminada.

- Si el usuario digita '2' en el menu principal, se presenta en la pantalla:



CREACION DE CONSULTAS
=====

DIGITE EL NOMBRE DE LA CONSULTA-->_

Fig.7.7.3

Si el nombre de la consulta es correcto y el archivo ya existe, aparece el mensaje:

CONSULTA YA EXISTE

SI INSISTE PIERDE LA CONSULTA ANTERIOR

DESEA CONTINUAR S/N-->_

Si digita 'N', regresa al menu principal.

El proceso de creacion solicita que se ingrese las lineas de la siguiente manera: muestra por pantalla '1>' para que se ingrese la primera linea, '2>' para que se ingrese la segunda linea, y asi sucesivamente hasta que el usuario ingrese una linea en blanco o hasta que la consulta tenga 10 lineas; siendo 49 el maximo de caracteres por linea.

Si la primera linea que ingresa el usuario esta en blanco, aparece un mensaje:

NO PUEDE CREAR UNA CONSULTA EN BLANCO

- Si el usuario digita '3' en el menu principal, en la pantalla se presentara:

```
MODIFICACION DE CONSULTAS
=====

DIGITE EL NOMBRE DE LA CONSULTA-->_
```

Fig.7.7.4

Si el nombre de la consulta es correcto pero el archivo no existe, aparece en la pantalla el mensaje:

ARCHIVO NO EXISTE

A continuacion aparece el siguiente menu:

```
MODIFICACION DE CONSULTAS
=====

1.- MODIFICA UNA LINEA
2.- ELIMINA UNA LINEA
3.- ACTIVA UNA LINEA BORRADA
4.- INSERTA UNA LINEA
5.- FIN DE MODIFICACION

DIGITE OPCION DESEADA->_
```

Fig.7.7.5

Siempre que la pantalla pide numero de linea, este debe ser menor que 11 ,si la linea que se ingresa esta en blanco aparece el mensaje:

LINEA XX NO EXISTE

'1' hace que en la pantalla se presente:

```
MODIFICACION DE UNA LINEA
=====
```

```
NUMERO DE LA LINEA->5
```

```
5> PJ:=Y%EDAD
   PL:=X%EDAD
```

Fig.7.7.6

Una vez que el numero de linea esta correcto, muestra la linea que va a ser modificada y aparece el cursor debajo del primer caracter de esa linea para que la nueva linea sea digitada.

```
ELIMINACION DE UNA LINEA
=====
```

```
NUMERO DE LA LINEA-> 5
```

```
5>PL:=X%EDAD
```

```
LINEA 5 ELIMINADA
```

```
PRESIONE <RETURN> PARA CONTINUAR
```

Fig.7.7.7

Cuando la eliminacion ha sido realizada, aparece el mensaje:

```
LINEA XX ELIMINADA
```

'3' hace que se muestre en la pantalla:

REACTIVACION DE UNA LINEA BORRADA
=====

NUMERO DE LA LINEA-> 2

2>SE:=R:NOMBRE=MATILDE

LINEA 2 YA ESTA ELIMINADA

Fig.7.7.8

Si el numero de linea que digita corresponde a una linea que no esta eliminada, aparece el mensaje:

LINEA 2 YA ESTABA ACTIVADA

'4' permitira:

INSERCIÓN DE UNA LINEA
=====

SE INSERTA LUEGO DE LA LINEA SENALADA

NUMERO DE LA LINEA-> 2

Fig.7.7.9

Si la consulta ya tiene 10 lineas se presenta el

mensaje:

SU CONSULTA YA TIENE 10 LINEAS
NO PUEDE INSERTAR

Si el numero de linea esta correcto, se presentara la siguiente pantalla:

```

      INSERCIÓN DE UNA LINEA
      =====
SE INSERTA LUEGO DE LA LINEA SENALADA
      NUMERO DE LA LINEA-> 2
      2>SA:=D:EDAD=23
      DIGITE LA LINEA QUE QUIERE INSERTAR
      3> PA:=J%NOMBRE
      LINEA 3 HA SIDO INSERTADA
  
```

Fig.7.7.10

Si la linea que se quiere insertar esta en blanco, se presenta en la pantalla el mensaje:

NO PUEDE INSERTAR LINEA EN BLANCO

Caso contrario aparece el mensaje:

LINEA XX HA SIDO INSERTADA

'5' hace que se presente nuevamente el menu principal (Ver Fig. 7.7.1).

- Si el usuario digita '4', en la pantalla se presenta:

```

      ** CONSULTA **
      =====

DIGITE EL NOMBRE DE LA CONSULTA--> AA
DESEA VER TODA LA CONSULTA S/N-> N
  
```

Fig.7.7.11

Si el archivo no existe aparece el mensaje:

ARCHIVO NO EXISTE

En caso de que el archivo si existe, pregunta si desea ver toda la consulta. Cuando la respuesta es 'N', se presenta la siguiente pantalla:

```

      ** CONSULTA **
      =====

DIGITE EL NOMBRE DE LA CONSULTA--> AA
DESEA VER TODA LA CONSULTA S/N-> N
      NUMERO DE LA LINEA-> 3
      3> PA:=J%NOMBRE

PRESIONE <RETURN> PARA CONTINUAR
  
```

Fig.7.7.12

Cuando la respuesta es 'S', la pantalla queda de la siguiente forma:

```

** CONSULTA **
=====

DIGITE EL NOMBRE DE LA CONSULTA--> AA

DESEA VER TODA LA CONSULTA S/N->S

1> SA:=M:NOMBRE=JORGE
2> JW:=A*B
3> PA:=J%NOMBRE
4> SB:=A:EDAD=21

PRESIONE <RETURN PARA CONTINUAR
```

Fig.7.7.13

- Si el usuario digita '5', se presenta la pantalla de las UTILITARIAS.

**8. ACCESO A LA BASE A TRAVES
DE UN PROGRAMA PASCAL**

8.1 INTRODUCCION

En este manual se describiran algunos de los procedimientos residentes en la Libreria del Sistema de la Base de Datos POLIREL; para permitir al programador PASCAL tener acceso a las relaciones de la Base.

En primer lugar se introducira al programador en la Operativa de la Unidad Polirel, se le mostrara como preparar el medio ambiente de trabajo, como iniciar un programa, como verificar el estado de las operaciones realizadas, como definir su formato de registro, como recuperar los datos y todas aquellas herramientas que el programador debe conocer antes de empezar a programar con Polirel. Cada procedimiento se describira con un ejemplo explicativo, el cual tendra una breve exposicion del objetivo del mismo, seguido por el programa y terminara con la explicacion de los puntos mas caracteristicos de cada programa.

Al final tendremos un ejemplo recapitulativo, el cual sera una demostracion practica de la capacidad de la Base de Datos; se definira un Sistema de Rol de Pagos de requerimientos minimos en el cual se mostrara la creacion de un archivo de rol y la emision de los reportes respectivos, esto se hara en conjuncion con el Generador de Reportes de la Base (POLIREP).

8.2 MEDIO AMBIENTE DE TRABAJO

Aquellos programadores que vayan a utilizar los procedimientos de la Unidad Polirel, deberán primeramente preparar el medio ambiente de trabajo adecuado para lo cual requiere:

- Un diskette con el Sistema Operativo de PASCAL (Apple 1).
- La Libreria del Sistema de la Base de Datos Relacional POLIREL
- Un diskette de trabajo para los archivos y relaciones de la Base.

El programador debera transferir la Libreria del Sistema de la Base de Datos al diskette con el Sistema Operativo PASCAL (System.Library), utilizando el FILER de este Sistema Operativo en donde ejecutara el comando TRANSFER de la siguiente forma:

Si el diskette con el Sistema Operativo esta en el Drive #1 y el diskette de la base en el Drive #2 el comando debe ser:

```
TRANSFER: #5:SYSTEM.LIBRARY
```

```
TO WHERE: #4:$
```

dejando que esta libreria del Sistema reemplace a la existente.

El usuario debera colocar el diskette con el sistema

operativo en el DRIVE #1 y el diskette con las relaciones de la base en el DRIVE #2. Los procedimientos de la unidad Polirel siempre buscaran las relaciones en el diskette del drive # 2, por lo tanto el usuario debera asegurarse de que sus relaciones siempre se encuentren en este drive.

8.3 COMO INICIAR UN PROGRAMA

Todo programa que invoque a los procedimientos de la Unidad Polirel deberán tener las siguientes declaraciones despues del enunciado del programa:

```
PROGRAM XXXXXXXX(INPUT,OUTPUT);
```

```
USES POLIREL;
```

```
CONST
```

```
TYPE
```

```
VAR
```

Esto indicara al compilador PASCAL que haga disponible todos los procedimientos de la Unidad Polirel, residente en la Libreria del Sistema. Si el usuario obtiene un error de compilacion en esta sentencia debera asegurarse que la Libreria del Sistema de la Base de Datos relacional esta disponible en el diskette con el Sistema Operativo PASCAL. Si ha instalado correctamente el sistema de libreria en su diskette de trabajo, al iniciar la ejecucion de su programa podra observar en la parte superior derecha de su pantalla algo semejante a:

```
POLIREL[?]
```

en donde ? es una sigla que ira variando de acuerdo a la operacion que la unidad este ejecutando (leer, grabar, salvar areas de memoria, verificar datos, accesar a las relaciones etc...).

8.4 CONDICION DE LAS OPERACIONES

Todos los procedimientos descritos en este manual estan condicionados a un estado de "listo" para dicha operacion, es decir, el resultado de la operacion realizada por el usuario puede haberse ejecutado exitosamente o pudo haberse cancelado debido a ciertas causas, para controlar todos estos posibles casos en la interface de la Unidad Polirel se ha definido:

OPCONDICION: INTEGER;

Esta variable entera global almacenara un valor, el cual reflejara el estado de la operacion realizada y su valor debera ser examinado por los usuarios de Polirel despues de haber realizado una invocacion a uno de sus procedimientos. Ejemplo:

```
BEGIN
.
.
.
LECSECTU('A',TABLA)
IF OPCONDICION <> 0 THEN RUTINAERROR;
.
.
.
END;
```

En este ejemplo se invoca el procedimiento LECSECTU para leer una tupla de la relacion 'A'; si la lectura de dicha tupla ha sido exitosa la variable OPCONDICION

almacenara un valor 0 (cero), caso contrario ha ocurrido algun error en la lectura pedida. El significado de todos y cada uno de los posibles valores que puede tomar la variable OFCONDICION se dan en el Apendice B de este manual.

8.5 DEFINICION DEL FORMATO DE REGISTRO DE TRABAJO

El punto mas importante y al cual se debe el exito del funcionamiento de la Unidad Polirel es la INTERFACE; es decir, la forma como los programas del usuario se comunican con los procedimientos de la Unidad; esta interface es fija por lo tanto el usuario esta obligado a seguir las instrucciones que se describen a continuacion para evitar que se produzcan resultados inesperados.

En la interface de la Unidad Polirel se ha declarado lo siguiente:

TYPE

:

FORMATO = PACKED ARRAY[0..10] OF STRING[7];

Esta declaracion sirve para que el usuario defina el formato de su registro de trabajo como se describira mas adelante y a su vez permitira que la Unidad reciba este formato con el mismo tipo de datos para lo cual se ha definido en los principales procedimientos:

[LECSECTU]

PROCEDURE [LECDIRTU] (RELACION: STRING;
[ACTUALTU] TABLATRIBUTOS:FORMATO);

Como se puede apreciar el parametro formal en los procedimientos de la Unidad (interface) son de tipo FORMATO, esto involucra que los parametros actuales en los programas del usuario sean del mismo tipo, es decir, el usuario en sus programas definira una o mas variables de tipo FORMATO. Ejemplo:

```
VAR
  TABLA:FORMATO;
```

Formato esta definido como un arreglo empaquetado de 11 ocurrencias, cada ocurrencia representa una columna determinada de una relacion, a excepcion de la ocurrencia cero (0).

La estructura de cada ocurrencia es la siguiente:

```
x xxxxxx
| -----
|      |
|      |
|      | -----> nombre de la columna
|      |
|      | -----> estrategia de la columna
|      |
|      | -----> estrategia de la columna
```

Ejemplo:

```
ECODIGO --> representa la columna denominada
           CODIGO y cuya estrategia es E(entero)
```

Este orden debe ser siempre respetado por el usuario que esta disenando su registro de trabajo, llenando cada una de estas ocurrencias (a partir de la ocurrencia 1) con la estrategia y el nombre de las columnas que desea procesar para dicha relacion, el orden en el que son

almacenados puede ser distinto al orden de definicion de la relacion, pero este orden debera seguir para la recuperacion de los datos respectivos.

Si al usuario le interesan todas las columnas de la relacion puede usarse lo anterior llenando el elemento cero (0) del arreglo con asteriscos, asi:

```
VAR
  TABLA:FORMATO;
  .
  .
  .
BEGIN
  .
  .
  TABLA[0]:='*****';
  .
  .
  .
END;
```

Por el contrario si al usuario le interesan solo determinadas columnas de la relacion, puede definir las llenando las ocurrencias del arreglo con los datos de las columnas deseadas, y finalizar con una entrada de asteriscos para indicar el fin de datos, asi:

```
TABLA[1]:='ECODIGO';
TABLA[2]:='SNOMBRE';
TABLA[3]:='CSEXO';
TABLA[4]:='RSUELDO';
TABLA[5]:='*****';
```

Este arreglo asi definido sera transmitido como parametro a los procedimientos LECSECTU, LECDIRTU,



ACTUALTU, en donde se verificara su contenido para asegurar el correspondiente tipo de datos y la correcta devolucion de los mismos.

Una vez que el usuario ha definido el formato de su registro de trabajo, podra ejecutar los procedimientos respectivos; en caso de no definir el formato de registro o definirlo pero con datos equivocados el usuario obtendra un OPCONDICION=94.

8.6 RECUPERACION DE DATOS DE LA INTERFACE DE LA UNIDAD

Hasta ahora hemos hablado de como decirle a la Unidad Polirel cual es nuestro registro de trabajo y hemos aprendido a definirlo; veremos ahora como la Unidad Polirel nos devuelve los datos solicitados.

En la interface de la unidad se han definido las siguientes reglas:

```
VAR
  ENPOL: PACKED ARRAY[0..10] OF INTEGER;
  CAPOL: PACKED ARRAY[0..10] OF CHAR;
  STPOL: PACKED ARRAY[0..10] OF STRING;
  REPOL: PACKED ARRAY[0..10] OF REAL;
```

Se ha definido un arreglo de 11 elementos por cada estrategia existente en la Base (E,C,S,R), de los cuales el elemento cero (0) es de uso exclusivo de la Unidad, en donde se almacenaran los valores de control para cada estrategia, por lo tanto el usuario no debera utilizarlos ni modificarlos para evitar resultados inesperados.

Como pueden apreciar cada arreglo representa al correcto tipo de datos para cada estrategia y es aqui donde el usuario recibira los datos solicitados dependiendo de la estrategia de los mismos, es decir, cada elemento del formato de registro definido por el usuario anteriormente tendra su dato en el arreglo

correspondiente dependiendo de la estrategia del mismo, en esta parte la posicion de cada elemento en el registro de trabajo definido por el usuario junto con su estrategia sera la clave para recuperar el dato requerido del arreglo correspondiente; asi, si el usuario ha definido en su registro de trabajo lo siguiente:

```

VAR
  TABLA:FORMATO;
  .
  .
  .
BEGIN
  .
  .
  .
  TABLA[1]:= 'ECODIGO';
  TABLA[2]:= 'SNOMBRE';
  TABLA[3]:= 'CSEXO';
  TABLA[4]:= 'RSUELDO';
  TABLA[5]:= '*****';

```

Los datos despues de ejecutar uno de los procedimientos (LECSECTU, LECDIRTU, ACTUALTU) se encontraron en las posiciones correspondientes de los arreglos definidos asi:

```

ECODIGO --> el dato se ha depositado en ENPOL[1]
SNOMBRE --> el dato se ha depositado en STPOL[2]
CSEXO    --> el dato se ha depositado en CAPOL[3]
RSUELDO  --> el dato se ha depositado en REPOL[4]

```

Observese las siguientes correspondencias en esta parte:

- Correspondencia de estrategia

'E' -- ENPOL[?];

'S' -- STPOL[?];

'C' --> CAPOL[?];

'R' --> REPOL[?];

- Correspondencia posicional entre el registro de trabajo y los arreglos de datos.

TABLA[1]:= 'ECODIGO' --> ENPOL[1]

·
·
·

TABLA[4]:= 'RSUELDO' --> REPOL[4]

^ ^



Estas dos reglas de correspondencia deberan ser seguidas obligatoriamente por los usuarios de la Unidad Polirel. Es responsabilidad absoluta del programador respetar estas relaciones para evitarse resultados inesperados ya que las ocurrencias de los arreglos sin correspondencia estan indefinidas.

Ahora si el usuario codifica en su programa:

TABLA[0]:= '*****';

para indicar que trabajara con todas las columnas de esa relacion, debera constatar con el Administrador de la Base de Datos para asegurarse el orden en el cual fueron definidas las columnas (Ver Definicion de Relaciones en el Manual del Sistema, Numeral 4.1) de dicha relacion con sus respectivas estrategias, para poder realizar una correcta recuperacion de datos.

8.7 PROGRAMAS DEMOSTRATIVOS

8.7.1 PROGRAMA DE ACCESO SECUENCIAL A LAS RELACIONES

El objetivo de este programa es permitir al usuario de la Unidad Polirel acceder a cualquier relacion de la base de datos; este hace posible visualizar todas las tuplas existentes (solo las activas) dentro de una relacion dada, una por una y en el orden en que fueron creadas.

El programa pedira al usuario el nombre de la relacion a acceder, el cual sera un solo caracter alfabetico, y cuyos archivos deben residir en el diskette de trabajo del usuario (drive # 2); el programa manipula todas las columnas de dicha relacion mostrandola luego en la pantalla con sus respectivos datos.

La lectura es secuencial hasta encontrar el fin de datos, el cual se lo identifica por un OPCION = 99 o por FINDATA = TRUE.

```

    1
(*$S+*)
PROGRAM SECUENCIAL;
(*****
(*)
(*) PROGRAMA DEMOSTRATIVO DE LECTURA SECUENCIAL *)
(*) DE RELACIONES, SE LEE TODAS LAS TUPLAS ACTI- *)
(*) VAS EN EL ORDEN EL QUE FUERON CREADAS UTILI- *)
(*) ZANDO EL MODULO "LECSECTU". *)
(*) F.E.J.L.*)
(*****)
USES POLIREL,COMMON;
VAR
    TABLA:FORMATO;
    I:INTEGER;
    CH:CHAR;
    STR1:STRING;
    ST1:STRING[2];

PROCEDURE CASESTRATEGIA(I:INTEGER);
VAR
    ST1:STRING[7];
BEGIN
    ST1:=TABLA[I];
    CASE ST1[1] OF
        'E': WRITELN (' ',ENPOL[I]);
        'R': WRITELN (' ',REPOL[I]:5:2);
        'C': WRITELN (' ',CAPOL[I]);
        'S': WRITELN (' ',STPOL[I]);
    END;
END;

BEGIN
    PAGE(OUTPUT);
    PROMPTAT(01,'** DEMOSTRACION DE ACCESO SECUENCIAL **');
    PROMPTAT(10,'** DIGITE NOMBRE DE RELACION-->');
    ST1:=' ';
    ST1[1]:=GETCHAR(ALFABETO);
    TABLA[0]:='*****';
    7
    2
    3
    4
    5
    6

```



```

    REPEAT
      LECSECTU(ST1,TABLA);
      IF (OPCONDICION <> 99) AND (OPCONDICION = 0)
      THEN
        BEGIN
          GOTOXY(00,17);
          FOR I:= 1 TO NUMCOLUMNAS DO
            BEGIN
              WRITELN('
              GOTOXY(00,I + 16);
              WRITE (TABLA[I], ' ');
              CASESTRATEGIA(I)
            END;
          END
        ELSE
          IF OPCONDICION <> 99 THEN
            BEGIN
              WRITELN ('** ERROR **',OPCONDICION);
              EXIT(PROGRAM)
            END;
          PROMPTAT(23,'<RETURN> PARA CONTINUAR');
          CH:=GETCHAR([CHR(13)]);
        UNTIL FINDATA;
      END.
  
```

Flowchart connections:

- Circle 8 points to the start of the REPEAT loop.
- Circle 9 points to the IF statement inside the REPEAT loop.
- Circle 10 points to the BEGIN block inside the IF statement.
- Circle 11 points to the END statement of the inner BEGIN block.
- Circle 12 points to the PROMPTAT statement in the ELSE block.
- Circle 13 points to the UNTIL FINDATA statement.

NOTAS DE PROGRAMACION

1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.

2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base; (*) La declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.

3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar.

4.- (**) Cuando el usuario especifica que trabajara con todas las columnas con la notacion del punto siete(7), la unidad devolvera en el registro de trabajo definido los nombres con sus

respectivas estrategias de todas las columnas existentes, en este punto se examina la estrategia de cada columna (posicion) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- El programa pide digitar el nombre de la relacion, el cual se lo obtiene con los procedimientos de la unidad Common.

6.- Funcion de la unidad Common que nos permite leer de consola un solo caracter; ALFABETO es una variable tipo SET predefinida en la unidad Polirel como ['A'...'Z', 'a'...'z'].

7.- Notacion que indica a la unidad Polirel que se trabajara con todas las columnas de la relacion dada, observese la definicion de dicha variable en VAR.

8.- Invocacion al procedimiento de lectura secuencial residente en la Unidad Polirel, la



relacion motivo de la lectura se ha almacenado en la variable ST1, TABLA se ha definido como el registro de trabajo, los datos de las tuplas existentes seran depositados en los arreglos respectivos (CAPOL, REPOL, STPOL, ENPOL) empezando por la tupla cero hasta la tupla N-1.

9.- Se debe siempre examinar el resultado de la operacion realizada, si el valor del OPCODE es igual a cero la operacion fue exitosa.

10.- NUMCOLUMNS (variable predefinida) almacenara despues de realizada la operacion el numero de columnas existentes dentro de la relacion, por lo tanto se condicionara a esta variable el proceso de las mismas.

11.- Si el OPCODE es diferente de 99 (fin de datos) entonces un error en la lectura ha ocurrido, se imprime el valor de OPCODE para detectar el error.

12.- Se detiene la ejecucion del programa para permitir al usuario visualizar todos los datos de la tupla leida, el proceso continuara cuando el

usuario presione <return>.

13.- El proceso principal se condiciona al fin de datos (FINDATA = TRUE), es decir, se leeran todas las tuplas activas existentes.

(*) El usuario puede obtener un mapa de la libreria del sistema con la utilitaria LIBMAP del sistema Operativo Pascal, para que pueda tener conocimiento de todas las variables, funciones y procedimientos disponibles en la unidad del sistema.

(**) Si el usuario decide no utilizar todas las columnas de su relacion puede definir las en la forma descrita anteriormente, por ejemplo:

```
BEGIN
```

```

TABLA[1] := 'SNOMBRE';
TABLA[2] := 'CESTADO';
TABLA[3] := 'RSUELDO';
TABLA[4] := 'ETEFEL';
TABLA[5] := '*****';
LECSECTU (ST1, TABLA);
IF OPCONDICION = 0 THEN
  BEGIN

```

```
END.
```

El usuario debe tomar en cuenta que el proceso de las tuplas ya no sera de 1 a NUMCOLUMNAS sino que el usuario debera utilizar otra variable que represente al numero de columnas que se estan tratando.

8.7.2 PROGRAMA DE ACCESO DIRECTO A LAS RELACIONES

El modulo de acceso secuencial LECSECTU descrito en el topico anterior puede convertirse en modulo de acceso directo alterando el valor del apuntador de registro corriente en el archivo AT, este apuntador de registro denominado TUPLANUMERO puede ser manejado por el usuario en su programa como se vera en el siguiente ejemplo.

El programa esta basado en el anterior con las respectivas modificaciones para pedir al usuario el numero de tupla a leer, las tuplas para el usuario estan numeradas desde 1 a N, fisicamente las tuplas estan numeradas desde 0 a N-1; el usuario continuara indefinidamente la lectura de tuplas digitando indistintamente numeros de tuplas entre 1 y NUMTUPLAS.

```

(**$+*)
PROGRAM LECDIREC;
(*****
(*)
(*) PROGRAMA DEMOSTRATIVO DE ACCESO DIRECTO A (*)
(*) LAS RELACIONES UTILIZANDO EL MODULO DE (*)
(*) LECTURA SECUENCIAL "LECSECTU", SE UTILIZA (*)
(*) LA POSICION FISICA DE LAS TUPLAS COMO CLA- (*)
(*) VE DE ACCESO. (*)
(*) F.E.J.L. *)
(*****)
USES POLIREL,COMMON;
VAR
  TABLA:FORMATO;
  I:INTEGER;
  CH:CHAR;
  STR1:STRING;
  ST1:STRING[2];

PROCEDURE CASESTRATEGIA(I:INTEGER);
VAR
  ST1:STRING[7];

BEGIN
  ST1:=TABLA[I];
  CASE ST1[1] OF
    'E': WRITELN (' ',ENPOL[I]);
    'R': WRITELN (' ',REPOL[I]:5:2);
    'C': WRITELN (' ',CAPOL[I]);
    'S': WRITELN (' ',STPOL[I]);
  END;
END;

BEGIN
  PROMPTAT(01,'** DEMOSTRACION DE ACCESO DIRECTO **');
  PROMPTAT(10,'** DIGITE NOMBRE DE RELACION-->');
  ST1:= ' ';
  ST1[1]:=GETCHAR(ALFABETO);
  TABLA[0]:= '*****';
  LECSECTU(ST1,TABLA);
  IF OPCONDICION <> 00 THEN
    BEGIN
      GOTOXY(0,23);
      WRITELN ('** ERROR **',OPCONDICION);
      EXIT(PROGRAM)
    END;
END;

```

```

      8
REPEAT
  REPEAT
    PROMPTAT(15,'NUMERO DE TUPLA ==>');
    GETSTRING(STR1,2,NUMEROS);
    STRINGENTERO(STR1,I);
    UNTIL I IN [1..NUMTUPLAS];
    TUPLANUMERO := I-1;
    LECSECTU(ST1,TABLA);
    IF (OPCONDICION = 0)
      THEN
        BEGIN
          GOTOXY(00,17);
          FOR I:= 1 TO NUMCOLUMNAS DO
            BEGIN
              WRITELN('
              GOTOXY(00,I + 16);
              WRITE (TABLA[I], ' ');
              CASESTRATEGIA(I)
            END;
          END
        ELSE
          IF OPCONDICION <> 99 THEN
            BEGIN
              GOTOXY(0,23);
              WRITELN ('** ERROR **',OPCONDICION);
              EXIT(PROGRAM)
            END;
          PROMPTAT(23,' ** OTRA TUPLA (S/N) **');
          UNTIL NOT YES;
        END.
      13

```

11

12

13

NOTAS DE PROGRAMACION

1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.

2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base; la declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.

3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar.

4.- Cuando el usuario especifica que trabajara con todas las columnas con la notacion del punto seis(6), la unidad devolvera en el registro de trabajo definido los nombres con sus respectivas

estrategias de todas las columnas existentes, en este punto se examina la estrategia de cada columna (posicion 1) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- El programa pide digitar el nombre, de la relacion, el cual se lo obtiene con los procedimientos de la unidad Common.

6.- Notacion que indica a la unidad Polirel que se trabajara con todas las columnas de la relacion dada, observese la definicion de dicha variable en VAR.

7.- Invocacion al procedimiento de lectura secuencial residente en la Unidad Polirel, la relacion motivo de la lectura se ha almacenado en la variable ST1, TABLA se ha definido como el registro de trabajo, los datos de las tuplas existentes seran depositados en los arreglos respectivos (CAPOL, REPOL, STPOL, ENPOL), esta

lectura se lo realiza para la validacion de la relacion solicitada.

8.- El programa pide al usuario el numero de la tupla a leer, entendiendose por numero de tupla a la posicion fisica de una tupla dada dentro de una relacion (sin considerar aquellas tuplas que han sido eliminadas).

9.- Se almacena el valor obtenido dentro de la variable TUPLANUMERO que trabaja como puntero corriente de tupla; se le resta una unidad para justificar la diferencia de notacion de tuplas logicas (1 a N) y tuplas fisicas (0 a n -1).

10.- Se debe siempre examinar el resultado de la operacion realizada, si el valor del OPCONDITION es igual a cero la operacion fue exitosa.

11.- NUMCOLUMNAS (variable predefinida) almacenara despues de realizada la operacion el numero de columnas existentes dentro de la relacion, por lo tanto se condicionara a esta variable el proceso de las mismas.

12.- Si el OPCODE es diferente de 99 (fin de datos) entonces un error en la lectura ha ocurrido, se imprime el valor de OPCODE para detectar el error.

13.- Se detiene la ejecución del programa para permitir al usuario visualizar todos los datos de la tupla leída, el programa da al usuario la opción de continuar con el proceso de lectura secuencial o de cancelarlo.

8.7.3 PROGRAMA DE ACCESO DIRECTO A LAS RELACIONES CON LA UTILIZACION DE INDICES INVERTIDOS Y DE CLASIFICACION

Este programa demostrara la utilizacion de los archivos invertidos y de clasificacion generados por las utilitarias de la base de Datos Polirel, el usuario entonces estara en capacidad de emplearlos de acuerdo a sus necesidades.

El programa contempla el proceso de archivos invertidos como de clasificacion. El programa identificara el tipo de archivo a utilizar por el nombre de la relacion, un archivo invertido tiene una 'C' en la segunda posicion de su nombre mientras un archivo de clasificacion contiene una 'S' en dicha posicion. Si el archivo a procesar es de indice invertido se le pedira al usuario el criterio de seleccion dependiendo de la estrategia del mismo, estrategia que sera dada por el usuario anteriormente.

Una vez identificado el criterio de seleccion se invocara al procedimiento LECDIRTU el cual leera todas las tuplas existentes para dicho criterio; en los dos casos el usuario debera almacenar la clave de busqueda en los arreglos adecuados como se explicara a continuacion.

```

1
(**$S+*)
PROGRAM DIRECTO;
(*****
(*)
(*) PROGRAMA DEMOSTRATIVO DE ACCESO DIRECTO A (*)
(*) LAS RELACIONES UTILIZANDO ARCHIVOS INVERTI- (*)
(*) DOS O DE CLASIFICACION, EL ARCHIVO A UTILI- (*)
(*) ZAR ES IDENTIFICADO POR EL SEGUNDO CARAC-- (*)
(*) TER DE SU NOMBRE. (*)
(*) F.E.J.L. (*)
(*****
USES POLIREL,COMMON; 2
VAR
  TABLA:FORMATO; 3
  I:INTEGER;
  CH:CHAR;
  SELECTION,STR1:STRING;

PROCEDURE CASESTRATEGIA(I:INTEGER);
VAR
  ST1:STRING[7];

BEGIN
  ST1:=TABLA[I]; 4
  CASE ST1[1] OF
    'E': WRITELN (' ',ENPOL[I]);
    'R': WRITELN (' ',REPOL[I]:5:2);
    'C': WRITELN (' ',CAPOL[I]);
    'S': WRITELN (' ',STPOL[I]);
  END;
END;

BEGIN
  PAGE(OUTPUT);
  PROMPTAT(01,'** DEMOSTRACION DE ACCESO DIRECTO **');
  REPEAT
    PROMPTAT(10,'** DIGITE NOMBRE DE RELACION-->');
    STR1:= ' ';
    GETSTRING(STR1,4,ALFABETO+NUMEROS); 5
    IF STR1[2] = 'C' THEN
      BEGIN
        PROMPTAT(17,'ESTRATEGIA DE LA CLAVE (E,R,S,C)');
        CH:= GETCHAR(['S','R','E','C']); 6
        PROMPTAT(18,'DIGITE LOS VALORES....');
        CASE CH OF

```

```

      6
      ↙
'E : BEGIN
    READLN(ENPOL[1]);
    SELECTION:= 'ENPOL';
END;
'S' : BEGIN
    READLN(STPOL[1]);
    SELECTION:= 'STPOL';
END;
'R' : BEGIN
    READLN(REPOL[1]);
    SELECTION:= 'REPOL';
END;
'C' : BEGIN
    CAPOL[1]:= GETCHAR(ALFABETO + NUMEROS);
    WRITELN;
    SELECTION:= 'CAPOL';
END;

END;
END
ELSE
    SELECTION:= '$SORT';
    7
    TABLA[0]:= '*****';
    REPEAT
        LECDIRTU(STR1,TABLA,SELECTION); 8
        PAGE(OUTPUT);
        IF (OPCONDICION = 0) 9
            THEN
                BEGIN
                    GOTOXY(00,17);
                    FOR I:= 1 TO NUMCOLUMNAS DO
                        BEGIN
                            WRITELN('
                                GOTOXY(00,I + 16);
                                WRITE (TABLA[I], ' ');
                                CASESTRATEGIA(I)
                            END;
                        END
                    END
                END
            ELSE
                IF OPCODECONDICION <> 100 THEN
                    BEGIN
                        WRITELN ('** ERROR **',OPCONDICION);
                        EXIT(PROGRAM)
                    END;
                    10
                    PROMPTAT(23,'** <RETURN> PARA CONTINUAR **');
                    CH:=GETCHAR (CHR(13));
                    UNTIL OPCODECONDICION = 100; (* FIN DE DATOS *)
                    PROMPTAT(23,'** OTRO VALOR (S/N) **');
                    UNTIL NOT YES;
                END.
                11

```



NOTAS DE PROGRAMACION

1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.

2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base; (*) La declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.

3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar.

4.- Cuando el usuario especifica que trabajara con todas las columnas con la notacion del punto siete(7), la unidad devolvera en el registro de trabajo definido los nombres con sus respectivas

estrategias de todas las columnas existentes, en este punto se examina la estrategia de cada columna (posicion 1) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- El programa pide digitar el nombre de la relacion, el cual se lo obtiene con los procedimientos de la unidad Common.

6.- Si el archivo a procesar es un archivo de indices invertidos el programa pide al usuario la estrategia de la clave y dependiendo de esta solicitara luego el valor del criterio de seleccion.

Observese como se lee cada valor (CASE ...) y se le almacena en la ocurrencia uno (1) del arreglo respectivo (segun su estrategia), notese ademas como se llena la palabra clave que identifica la posicion del criterio de seleccion utilizado.
ejemplo:

```
SELECTION := 'CAPOL':
```

Si se trata de un archivo de clasificacion no se pedira criterio de seleccion pero debemos indicarle al procedimiento LECDIRTU la clase de archivo a procesar con la notacion :

SELECTION := '\$SORT';

7.- Notacion que indica a la unidad Polirel que se trabajara con todas las columnas de la relacion dada, observese la definicion de dicha variable en VAR.

8.- Invocacion al procedimiento de lectura directa, el nombre de la relacion se ha almacenado en STR1, este debe ser de cuatro caracteres (ver formato de archivos invertidos y de clasificacion en este manual), TABLA identifica el diseno de registro definido por el usuario; SELECTION almacenara el valor que identifica la posicion (el arreglo) en donde ha sido depositado el valor del criterio de seleccion (solo para archivos invertidos) o especifica que se trata de un archivo de clasificacion

9.- Se debe siempre examinar el resultado de la operacion realizada, si el valor del OPCONDICION

es igual a cero la operacion fue exitosa.

10.- El proceso continua leyendo tupla a tupla en el orden en que se encuentran descritos en el archivo de indices invertidos o el archivo de clasificacion hasta detectar el fin de datos del mismo (OPCONDICION = 100).

11.- Cuando se esta procesando un archivo de indices invertidos puede leerse varios grupos de tuplas bajo diferentes criterios de seleccion, si el usuario no desea leer mas tuplas puede terminar el proceso.

8.7.4 PROGRAMA DE ACTUALIZACION DE TUPLAS

Este programa demostrara al usuario la forma de actualizar los archivos (relaciones) de la Base de Datos.

Antes de actualizar una tupla esta tiene que ser direccionada por el apuntador de tupla corriente (TUPLANUMERO), para lo cual es necesario que la tupla sea leida ya sea utilizando el modulo de lectura secuencial "LECSECTU" o el modulo de lectura directa "LECDIRTU", en el momento en que la tupla se encuentra en memoria sus valores pueden ser alterados; el diseno de registro definido en la lectura rige para la actualizacion, es decir, debera actualizarse todas las columnas leidas.

El proceso de actualizacion se lo realiza justamente despues de la lectura, esto debera ser tomado en consideracion para evitarse resultados impredecibles.

La metodologia usada requiere que todo proceso de actualizacion de tuplas termine con una sentencia FINPROCESO para mantener la integridad de los datos en la Base.

1

```
(**$S+**)
PROGRAM ACTUALIZA;
(*****)
(*
(* PROGRAMA DEMOSTRATIVO DE ACTUALIZACION DE
(* TUPLAS UTILIZANDO EL MODULO DE LECTURA SE-
(* CUENCIAL DE TUPLAS "LECSECTU" Y EL MODULO
(* DE ACTUALIZACION "ACTUALTU".
(*
(* F.E.J.L.*)
(*****)
```

2

```
USES POLIREL, COMMON;
```

VAR

3

```
TABLA:FORMATO;
I:INTEGER;
CH:CHAR;
ST1:STRING;
ST1:STRING[2];
```

```
PROCEDURE CASESTRATEGIA(I:INTEGER);
```

VAR

4

```
ST1:STRING[7];
```

BEGIN

```
ST1:=TABLA[I];
```

```
CASE ST1[1] OF
```

```
'E': WRITELN (' ',ENPOL[I]);
```

```
'R': WRITELN (' ',REPOL[I]:5:2);
```

```
'C': WRITELN (' ',CAPOL[I]);
```

```
'S': WRITELN (' ',STPOL[I]);
```

```
END;
```

```
END;
```

```
PROCEDURE READDATA(I:INTEGER);
```

VAR

5

```
ST1:STRING[7];
```

BEGIN

```
ST1:=TABLA[I];
```

```
CASE ST1[1] OF
```

```
'E': READLN (ENPOL[I]);
```

```
'R': READLN (REPOL[I]);
```

```
'C': BEGIN
```

```
CAPOL[I]:=GETCHAR([' ','.',' ']);
```

```
WRITELN;
```

```
END;
```

```
'S': READLN (STPOL[I]);
```

```
END;
```

```
END;
```

PROCEDURE LEEDATO;

6

BEGIN

REPEAT

PROMPTAT(15,'NUMERO DE TUPLA A ACTUALIZAR..');

GETSTRING(STR1,2,NUMEROS);

STRINGENTERO(STR1,I);

UNTIL I IN [1..NUMTUPLAS];

TUPLANUMERO:=I - 1;

LECSECTU(ST1,TABLA);

IF (OPCONDICION = 0)

THEN

BEGIN

FOR I:= 1 TO NUMCOLUMNAS DO

BEGIN

PROMPTAT(I+16,' ');

GOTOXY(00,I + 16);

WRITE (TABLA[I],' ');

CASESTRATEGIA(I)

END;

END

ELSE

BEGIN

WRITELN ('** ERROR **',OPCONDICION);

READLN;

EXIT(PROGRAM)

END;

END;

BEGIN

PAGE(OUTPUT);

PROMPTAT(01,'* PROGRAMA DE ACTUALIZACION DE TUPLAS *');

PROMPTAT(10,'** NOMBRE DE RELACION --> ');

ST1:=' ';

ST1[1]:=GETCHAR(ALFABETO);

TABLA[0]:='*****';

LECSECTU(ST1,TABLA);

IF OPCONDICION <> 0 THEN

BEGIN

PROMPTAT(22,' ** ERROR EN RELACION **');

WRITELN(' OP = ',OPCONDICION,

' ** PRESIONE <RETURN> **');

CH:= GETCHAR([CHR(13)]);

END;

```

WHILE TRUE DO
  BEGIN
    LEEDATO;
    PROMPTAT(15,'');
    GOTOXY(00,15);
    WRITE('DESEA ACTUALIZARLA...(S/N) ?');
    IF YES THEN
      BEGIN
        PROMPTAT(15,'DIGITE NUEVOS VALORES.....');
        GOTOXY(00,17);
        FOR I:= 1 TO NUMCOLUMNAS DO
          BEGIN
            WRITELN('');
            GOTOXY(00,I + 16);
            WRITE (TABLA[I],');
            READDATA(I);
            END;
            ACTUALTU(ST1,TABLA);
            IF OPCONDICION <> 0 THEN
              BEGIN
                WRITELN(' ** ACTUALIZACION NO EXITOSA ** OP= ',
                  OPCONDICION);
                PROMPTAT(23,' ** PRESIONE <RETURN> ** ');
                CH:= GETCHAR([CHR(13)]);
              END;
            END
          END
        ELSE
          BEGIN
            PROMPTAT(23,'** DESEA CONTINUAR...(S/N) ?');
            IF NOT YES THEN
              BEGIN
                FINPROCESO;
                EXIT(PROGRAM);
              END;
            END;
          END;
        END;
      END.

```

Diagram annotations: A circle labeled 10 points to the line 'PROMPTAT(15, 'DIGITE NUEVOS VALORES.....');'. A circle labeled 11 points to the line 'READDATA(I);'. A circle labeled 12 points to the line 'ACTUALTU(ST1, TABLA);'. A circle labeled 13 points to the line 'FINPROCESO;'.

NOTAS DE PROGRAMACION

1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.

2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base; la declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.

3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar.

4.- Cuando el usuario especifica que trabajara con todas las columnas con la notacion del punto (7), la unidad devolvera en el registro de trabajo definido los nombres con sus respectivas

estrategias de todas las columnas existentes, en este punto se examina la estrategia de cada columna (posicion 1) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- Los datos son leidos dentro de los arreglos respectivos de acuerdo a su estrategia (correspondencia de estrategia) y ademas en las posiciones adecuadas a la definicion de las columnas en la relacion (correspondencia posicional).

6.- Este procedimiento hace posible visualizar una tupla determinada, el metodo empleado es el descrito en el programa de Lectura Directa; esta lectura de tupla siempre debe ser realizada antes de ejecutar el procedimiento de actualizacion.

7.- Notacion que indica a la unidad Polirel que se trabajara con todas las columnas de la relacion dada, observese la definicion de dicha variable en



VAR.

8.- Invocacion al procedimiento de lectura secuencial residente en la Unidad Polirel, la relacion motivo de la lectura se ha almacenado en la variable ST1, TABLA se ha definido como el registro de trabajo, la lectura se lo hace con el proposito de verificar la relacion dada.

9.- Se debe siempre examinar el resultado de la operacion realizada, si el valor del OPCONDICION es igual a cero la operacion fue exitosa.

10.- El usuario tiene la oportunidad de determinar si la tupla leida se actualiza o no.

11.- Si el usuario ha decidido actualizar la tupla previamente leida, se le pide digitar los nuevos valores para todas las columnas existentes, el procedimiento READDATA ingresara los datos como se describio en el punto (5). Observese la posicion de la lectura de los nuevos datos inmediatamente despues de la lectura de la tupla y antes de la invocacion al procedimiento ACTUALTU.

12.- Cuando los datos han sido ingresados correctamente se invocara al procedimiento ACTUALTU para que proceda a reemplazar los nuevos valores, ST1 identifica la relacion, TABLA es el diseno de registro definido por el usuario en la lectura de la tupla.

13.- Todo proceso de actualizacion de tuplas finalizara con la sentencia FINPROCESO, para mantener la integridad de los datos contenidos en la Base. Obsérvese la posición de FINPROCESO, justo antes de finalizar el programa esta debe ser la posición adecuada dentro del programa.

Por lo tanto el esquema del proceso de actualizacion de tuplas que el usuario debe seguir es el siguiente:

```
BEGIN
.
.
LECSECTU (relacion,tabela-de-atributos)
IF OPCONDITION <> 0 THEN ...
.
aqui se ingresaran los nuevos datos
.
ACTUALTU(relacion);
IF OPCONDITION <> 0 THEN ...
.
.
FINPROCESO;
END.
```

8.7.5 PROGRAMA DE ELIMINACION DE TUPLAS

Este programa demostrara al usuario el procedimiento de eliminacion de tuplas no deseadas o aquellas que han dejado de ser utiles para la Base de Datos.

La secuencia de procedimientos sera seguida siempre como se describira mas adelante, es decir, primero se leera la tupla respectiva y luego se la eliminara; el programa utiliza el metodo de acceso directo a una tupla dando su posicion fisica dentro de la relacion como fue descrito en un programa anterior. Una vez que el usuario tiene en memoria una tupla determinada el programa pedira la opcion de eliminarla, en caso de tener una respuesta afirmativa a esta interrogacion se invocara al procedimiento ELIMINTU el cual pondra el estado correspondiente a dicha tupla.

El proceso continua indefinidamente hasta que el usuario decida terminarlo.

La metodologia usada requiere que todo proceso de actualizacion de tuplas termine con una sentencia FINPROCESO para mantener la integridad de los datos en la Base.

1

```
(**$S+*)
PROGRAM ELIMINA;
(*****)
(*                                     *)
(*  PROGRAMA DEMOSTRATIVO DE ELIMINACION DE TU- *)
(*  PLAS UTILIZANDO LOS MODULOS DE LECTURA SE- *)
(*  CUENCIAL "LECSECTU" Y DE ELIMINACION DE TU- *)
(*  PLAS "ELIMINTU". *)
(*                                     F.E.J.L. *)
(*****)
```

2

```
USES POLIREL,COMMON;
```

VAR

3

```
TABLA:FORMATO;
I:INTEGER;
CH:CHAR;
STR1:STRING;
ST1:STRING[2];
```

```
PROCEDURE CASESTRATEGIA(I:INTEGER);
```

VAR

ST1:STRING[7];

BEGIN

ST1:=TABLA[I];

CASE ST1[I] OF

'E': WRITELN (' ',ENPOL[I]);

'R': WRITELN (' ',REPOL[I]);

'C': WRITELN (' ',CAPOL[I]);

'S': WRITELN (' ',STPOL[I]);

END;

END;

BEGIN

PROMPTAT(01,'** DEMOSTRACION DE ACCESO DIRECTO **');

PROMPTAT(10,'** DIGITE NOMBRE DE RELACION-->');

ST1:= ' ';

ST1[I]:=GETCHAR(ALFABETO);

TABLA[0]:='*****';

LECSECTU(ST1,TABLA);

IF OPCONDICION <> 00 THEN

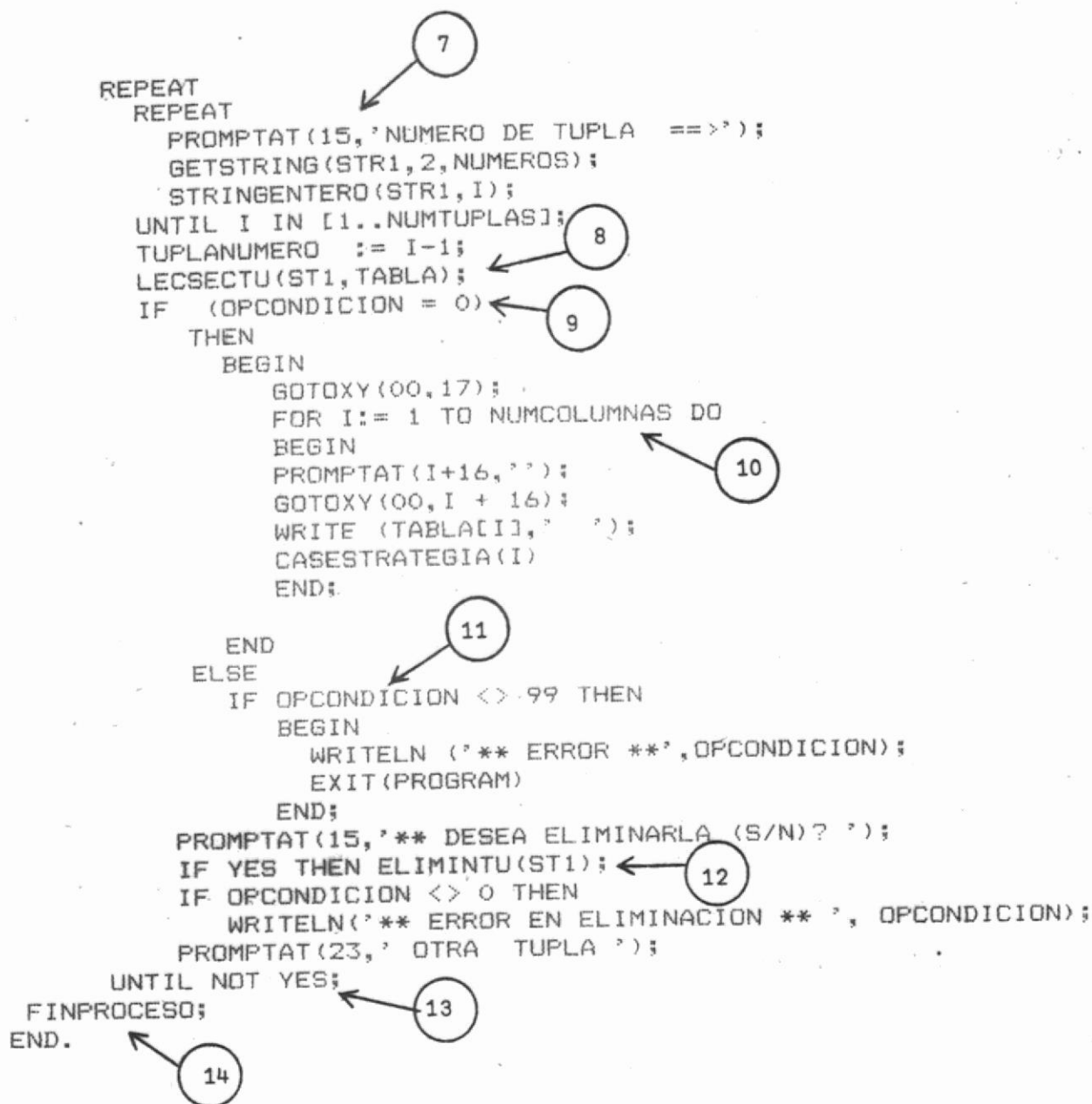
BEGIN

WRITELN ('** ERROR **',OPCONDICION);

EXIT(PROGRAM)

END;

6



NOTAS DE PROGRAMACION

1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.

2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base. La declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.

3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar.

4.- Cuando el usuario especifica que trabajara con todas las columnas con la notacion del punto (6), la unidad devolvera en el registro de trabajo definido los nombres con sus respectivas

estrategias de todas las columnas existentes, en este punto se examina la estrategia de cada columna (posicion 1) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- El programa pide digitar el nombre de la relacion, el cual se lo obtiene con los procedimientos de la unidad Common, se invoca al procedimiento de lectura para leer la primera tupla, ademas se lo realiza con propositos de validacion de la relacion.

6.- Notacion que indica a la unidad Polirel que se trabajara con todas las columnas de la relacion dada, observese la definicion de dicha variable en VAR.

7.- Se pide al usuario el numero de tupla a eliminar, el cual se almacena en la variable TUPLANUMERO para accesarla en forma directa.

8.- Invocacion al procedimiento de lectura secuencial residente en la Unidad Polirel, la relacion motivo de la lectura se ha almacenado en la variable ST1, TABLA se ha definido como el registro de trabajo, la lectura se lo hace con el proposito de que el usuario se asegure de que la tupla a eliminar es la correcta.

9.- Se debe siempre examinar el resultado de la operacion realizada, si el valor del OPCONDICION es igual a cero la operacion fue exitosa.

10.- NUMCOLUMNAS (variable predefinida) almacenara despues de realizada la operacion el numero de columnas existentes dentro de la relacion, por lo tanto se condicionara a esta variable el proceso de las mismas.

11.- Si el OPCONDICION es diferente de 99 (fin de datos) entonces un error en la lectura ha ocurrido, se muestra en pantalla el valor de OPCONDICION.

12.- Se detiene la ejecucion del programa para permitir al usuario visualizar todos los datos de

la tupla leída, esto permitira decidir si es una tupla correcta a eliminar, en caso de ser así se invocara al procedimiento ELIMINTU, el nombre de la relacion se ha almacenado en ST1, observese que se trata de la misma relacion de la lectura previa, la tupla eliminada sera aquella apuntada por TUPLANUMERO.

El resultado de la operacion realizada siempre debe ser verificado.

13.- El proceso continuara indefinidamente hasta que el usuario decida terminar.

14.- Todo proceso de eliminacion de tuplas debe terminar con una sentencia FINPROCESO para mantener la integridad de los datos contenidos en la Base; observese la posicion de esta sentencia justo antes de terminar el programa, esta debe ser la posicion adecuada en los programas del usuario. Por lo tanto el esquema del proceso de eliminacion de tuplas que el usuario debe seguir es el siguiente:

BEGIN

.
.
LECSECTU (relacion, tabla-de-atributos)
IF OPCONDICION <> 0 THEN ...

.
ELIMINTU(relacion);
IF OPCONDICION <> 0 THEN ...

.
FINPROCESO;
END.

8.7.6 PROGRAMA DE ADICION DE TUPLAS

Este programa demostrara al asuario el procedimiento de adicion de nuevas tuplas a las existentes dentro de una relacion dada; el programa tratara de visualizar este proceso leyendo una tupla (generalmente la ultima ingresada) y dejando que el usuario decida ingresar mas tuplas.

La lectura de tuplas realizada en este programa no es necesaria ya que se lo hace unicamente con el proposito de visualizar la ultima tupla insertada (a excepcion de la primera vez que se lee la primera tupla).

Los datos deben ser depositados en los arreglos correspondientes y los cuales deben ser iguales en numero a las columnas de la relacion, el usuario debera asegurarse de la correcta carga de las ocurrencias de los arreglos para evitar que estos queden indefinidos y asi no obtener resultados impredecibles.

La metodologia usada requiere que todo proceso de actualizacion de tuplas termine con una sentencia FINPROCESO para mantener la integridad de los datos en la Base.

1

```

(**$+**)
PROGRAM ADICIONA;
(*****
(*)
(*) DEMOSTRACION DE ADICION DE TUPLAS A UNA (*)
(*) RELACION DADA UTILIZANDO EL PROCEDIMIENTO (*)
(*) "ADICIOTU". (*)
(*) F.E.J.L.*
(*****)

```

2

```

USES POLIREL, COMMON;

```

3

```

VAR
  TABLA: FORMATO;
  I: INTEGER;
  CH: CHAR;
  STR1: STRING;
  ST1: STRING[2];

```

```

PROCEDURE CASESTRATEGIA(I: INTEGER);
VAR
  ST: STRING[7];

```

4

```

BEGIN
  ST:=TABLA[I];
  CASE ST[1] OF
    'E': WRITELN (' ', ENPOL[I]);
    'R': WRITELN (' ', REPOL[I]:5:2);
    'C': WRITELN (' ', CAPOL[I]);
    'S': WRITELN (' ', STPOL[I]);
  END;
END;

```

5

```

PROCEDURE READDATA(I: INTEGER);
VAR
  ST: STRING[7];

```

```

BEGIN
  ST:=TABLA[I];
  CASE ST[1] OF
    'E': READLN (ENPOL[I]);
    'R': READLN (REPOL[I]);
    'C': BEGIN
      CAPOL[I]:=GETCHAR(['...']);
      WRITELN;
    END;
    'S': READLN (STPOL[I]);
  END;
END;

```

PROCEDURE LEEDA10;

BEGIN

PAGE(OUTPUT);

TABLA[0]:='*****';

LECSECTU(ST1,TABLA);

IF OPCONCONDICION = 0

THEN

BEGIN

GOTOXY(00,05);

FOR I:= 1 TO NUMCOLUMNAS DO

BEGIN

PROMPTAT(I + 04,' ');

GOTOXY(00,I + 04);

WRITE (TABLA[I], ' ');

CASESTRATEGIA(I)

END;

END

ELSE

IF (OPCONCONDICION <> 00) THEN

BEGIN

WRITELN ('** ERROR **',OPCONCONDICION);

READLN;

EXIT(PROGRAM)

END;

END;

BEGIN

PAGE(OUTPUT);

PROMPTAT(01,'** DEMOSTRACION DE ADICION DE TUPLAS **');

PROMPTAT(10,'** NOMBRE DE RELACION --> ');

ST1:=' ';

ST1[1]:=GETCHAR(ALFABETO);

```

WHILE TRUE DO
  BEGIN
    LEEDATO;
    GOTOXY(00,15);
    WRITE('DESEA INSERTAR UNA NUEVA TUPLA ?');
    IF YES THEN
      BEGIN
        PROMPTAT(15,'DIGITE VALORES..... ');
        GOTOXY(00,17);
        FOR I:= 1 TO NUMCOLUMNAS DO
          BEGIN
            WRITELN('
            GOTOXY(00,I + 16);
            WRITE (TABLA[I],' ');
            READDATA(I); ← 8
            END;
            ADICIONTU(ST1); ← 9
            TUPLANUMERO:= NUMTUPLAS + 1;
            WRITELN( N= ,NUMTUPLAS : ← 7
          END
        ELSE
          BEGIN
            FINPROCESO; ← 10
            EXIT(PROGRAM)
          END;
        IF OPCONDICION <> 0 THEN
          BEGIN
            WRITELN(' ** ADICION NO EXITOSA OP = ',OPCONDICION);
            PROMPTAT(23,' ** PRESIONE <RETURN> ** ');
            CH:= GETCHAR([CHR(13)]);
          END;
        END;
      END.

```

NOTAS DE PROGRAMACION

- 1.- Opcion de SWAPPING, esta opcion deberia ser usada siempre que trabajemos con unidades. Esto hace que el programa tenga mas memoria principal disponible en el momento de la compilacion.
- 2.- Declaracion que siempre debe estar presente en todo programa que utilice la Unidad Polirel, esto permitira invocar cualquiera de los procedimientos disponibles para la manipulacion de las relaciones de la base; la declaracion COMMON despues de POLIREL indicara que este programa utilizara los procedimientos residentes en dicha unidad tambien.
- 3.- El usuario definira una o varias declaraciones de este tipo para describir su registro de trabajo, esto se lo debe hacer para enlazar los parametros actuales y parametros formales de los procedimientos a utilizar..
- 4.- Cuando el usuario especifica que trabajara con todas las columnas, la unidad devolvera en el registro de trabajo definido los nombres con sus respectivas estrategias de todas las columnas

existentes, en este punto se examina la estrategia de cada columna (posicion 1) para imprimir el dato respectivo. Los datos han sido dejados por la Unidad Polirel en los arreglos ENPOL, REPOL, CAPOL, STPOL de acuerdo a la estrategia declarada por el usuario en la definicion de su registro de trabajo.

5.- Los datos son leidos dentro de los arreglos respectivos de acuerdo a su estrategia (correspondencia de estrategia) y ademas en las posiciones adecuadas a la definicion de las columnas en la relacion (correspondencia posicional). El numero de datos a ingresar es igual al numero de columnas definidas para dicha relacion, por lo tanto no se necesita disenio de registro ya que este fue fijado al definir la relacion.

6.- Este procedimiento hace posible visualizar la ultima tupla ingresada en la relacion, la lectura de la misma es opcional y no forma parte del proceso de adiccion de tuplas.

7.- Esta notacion hara que el procedimiento de

lectura secuencial accese la ultima tupla ingresada.

8.- Si el usuario decide insertar una nueva tupla, los datos de todas las columnas existentes deben ser leidos (ver punto 5); es importante que el usuario ingrese datos a todas las columnas ya que podrian quedar columnas indefinidas, puesto que los arreglos pueden contener datos de adiciones anteriores o datos inicializados por la unidad (Datos nulos no son permitidos).

9.- Cuando todos los datos de las columnas existentes han sido colocados en los arreglos respectivos (aplicando la regla de correspondencia de estrategia y posicional), se invoca al procedimiento ADICIOTU el cual adicionara la tupla a continuacion de la ultima existente en el archivo. Recalcamos que la lectura a la tupla no es necesaria, se lo hace con el proposito de visualizar las tuplas recién ingresadas por el usuario, ademas sirve de guia en el orden y estrategia de cada columna en el momento de leer los nuevos datos, en caso de no realizar una lectura previa el usuario debera consultar al

Administrador de la Base de Datos el orden en que fueron definidas las columnas de la relacion y sus respectivas estrategias, para asi poder realizar una adicion correcta.

10.- Si no desea insertar mas tupla el programa termina con una sentencia FINPROCESO, para mantener la integridad de los datos mantenidos en la Base.

Por lo tanto el esquema del proceso de adicion de tuplas que el usuario debe seguir es el siguiente:

BEGIN

.

(* Lectura opcional *)

LECSECTU (relacion, tabla-de-atributos)

IF OPCONDICION <> 0 THEN ...

.

ADICIOTU(relacion);

IF OPCONDICION <> 0 THEN ...

.

FINPROCESO;

END.

9. GENERADOR DE REPORTES
(POLIREP)

9.1 INTRODUCCION

En este manual se describirán los diferentes procesos que deberá seguir el usuario para grabar los diversos formatos de los listados que el desee realizar.

Cada modulo de POLIREP sera explicado por medio de pantallas y tratando de hacer que el usuario pueda guiarse para la definicion de sus formatos.

Con el ejemplo de un sistema de Rol de Pagos, que se presentara, el usuario podra comprender mas facilmente las utilidades que le puede brindar POLIREP.



9.2 PROCESO GENERAL

Al elegir el usuario la opción '4' del menú principal de la base POLIREL va a aparecer la siguiente pantalla:

```

** GENERADOR DE REPORTES **
===== == =====

P O L I R E P

VERSION 1.0.

1.- CREACION DE FORMATOS
2.- CONSULTA DE FORMATOS
3.- CORRECCION DE FORMATOS
4.- EJECUCION DE FORMATOS
5.- ELIMINACION DE FORMATOS
6.- FIN DE SESION.

DIGITE LA OPCION DESEADA: x
```

Fig. 9.2.1

En la fig. 9.2.1. podemos ver la pantalla que presenta POLIREP al ser invocado. De acuerdo a la opción escogida por el usuario se presentaran las diferentes pantallas de cada opción. Cuando el usuario escoge la opción '6' se le presentara la siguiente pantalla:

** GENERADOR DE REPORTES **
=====

FIN DE SESION

PARA CONTINUAR INSERTE
DISKETTE BASE-1

PRECIONE <RETURN> PARA CONTINUAR

Fig. 9.2.2

Una vez que el usuario inserta el diskette de acuerdo a la Fig. 9.2.2 se retornara al menu principal de la base.

9.3 CREACION DE FORMATOS

Este modulo permite al usuario generar un formato para emitir un listado.

Al digitar la opcion 1 en el menu principal de POLIREP el usuario va a obtener la siguiente pantalla:

```
*** CREACION DE FORMATOS ***  
-----  
  
NOMBRE DEL FORMATO: FTO1  
=====
```

LISTA 1 O 2 RELACIONES : 1

ESTA CORRECTO S/N ? S

ERROR!!

Fig. 9.3.1.

El usuario debera digitar el nombre del formato que desea grabar, o presionar <RETURN> si desea regresar al menu principal de POLIREP.

Ademas debera definir el # de relaciones que desea listar, este # se lo considera como el tipo de listado a realizar. Tipo 1, sera el listado de 1 relacion y el tipo 2, sera el listado de 2 relaciones.

En el momento en que el usuario digita un nombre de formato POLIREP creara 3 archivos para ese formato.

En la fig. 9.3.1. se podran presentar los siguientes

errores:

ERROR!! FORMATO DEBE EMPEZAR POR LETRA

El primer caracter del nombre del formato siempre debera ser una letra.

ERROR!! EN ARCHIVOS, NO ESPACIO (FORMATO)

Al momento de crear el 1er. archivo, que es el de formato, no se encontro espacio en el disco, el usuario debera formatear otro diskette para crear sus formatos. En este diskette debera tener ademas las relaciones con las que va a trabajar el formato.

ERROR!! EN ARCHIVOS, NO ESPACIO (CABEC.)

Este error se le presentara al momento en que POLIREP vaya a crear el 2do. archivo, debe formatear otro diskette.

ERROR!! EN ARCHIVOS, NO ESPACIO (DETALLE)

Este error se presentara al crear el 3er. archivo de formato.

Una vez que ya se ha definido el tipo de listado, de acuerdo a esto se seguiran solicitando los datos.

Existen 3 procesos principales que se van a ejecutar en el siguiente orden:

- 1.- Creacion de datos de formato general.
- 2.- Creacion de datos de cabeceras
- 3.- Creacion de datos de detalle.

Cada uno de estos procesos se van a ir presentando en diferentes pantallas y van a ir solicitando los datos que se requieren de acuerdo al tipo de listado.

9.3.1 CREACION DE DATOS DE FORMATO GENERAL.-

Los datos de formato general son los que representan mayor importancia en el formato, porque de acuerdo a estos datos se va a realizar la ejecucion del listado.

El usuario debe tomar muy en cuenta el tipo de listado que ha definido y de acuerdo a eso se le van a ir pidiendo los datos.

9.3.1.1 CREACION DEL FORMATO GENERAL DE 1 RELACION.-

Cuando el listado es tipo 1 (listado de 1 relacion) el usuario debera definir que tipo de lectura desea realizar con la relacion que va a listar. Tiene 3 opciones a elegir y de acuerdo a la opcion que elija se le van a pedir los datos.

N : Normal, desea hacer una lectura secuencial.

I : Indice Invertido, desea que se lea la relacion a traves de un indice invertido.

C : Clasificacion, desea que se lea la relacion a traves de un archivo de clasificacion.

Lectura NORMAL (Tipo N)

**** CREACION DE FORMATO GENERAL ****

LINEAS POR PAGINA (1...80) : 66
 CARACTERES POR LINEA(1..132): 132
 # LINEAS CABECERAS(1....7) : 3
 # LINEAS DETALLE (1....3) : 1
 RELACION A LISTARSE : A
 LEE X INDICE,CLASIF,NORMAL : N

ESTA TODO CORRECTO S/N? S

ERROR!!

Fig. 9.3.1.1.1

En la fig. 9.3.1.1.1 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Este error se presenta cuando en alguno de los datos solicitados por pantalla no se ha respetado los rangos establecidos; así, líneas por pagina puede ser maximo 80, si se digita mas de ese valor se presentara este error.

ERROR!! RELACION NO EXISTE

Cuando la relacion que se desea listar no existe. Este error retorna el control al menu principal de POLIREP.

Lectura por Indice Invertido (Tipo I)

** CREACION DE FORMATO GENERAL **

```

LINEAS POR PAGINA (1...80) : 66
CARACTERES POR LINEA(1..132): 132
# LINEAS CABECERAS(1....7) : 3
# LINEAS DETALLE (1....3) : 1
RELACION A LISTARSE : A
LEE X INDICE,CLASIF,NORMAL : I
COLUMNA DE INDICE : 8
CRITERIO 1 -> ANACOX
CRITERIO 2 ->

```

ESTA TODO CORRECTO S/N? S

ERROR!!

Fig. 9.3.1.1.2

En la fig. 9.3.1.1.2 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO .

Este error se presenta, cuando en alguno de los datos solicitados por pantalla no se ha respetado los rangos establecidos; así, líneas por pagina puede ser maximo 80, si se digita mas de ese valor se presentara este error.

ERROR!! RELACION NO EXISTE

Cuando la relacion a listarse no existe. Este error retorna el control al menu

principal de POLIREP.

ERROR!! COLUMNA NO EXISTE

Este error se presenta cuando la columna de indice no existe en la relacion a listarse.

ERROR!! COLUMNA NO TIENE INDICE INVERT.

Cuando la columna de indice no tiene generado un indice invertido para ella.

DEBE HABER POR LO MENOS 1 CRITERIO

Para este tipo de lectura se requiere por lo menos poner 1 criterio para que se pueda efectuar la misma.

Lectura por Archivo de Clasificacion
(Tipo C)

** CREACION DE FORMATO GENERAL **

```

-----
LINEAS POR PAGINA (1...80) : 66
CARACTERES POR LINEA(1..132): 132
# LINEAS CABECERAS(1....7) : 3
# LINEAS DETALLE (1....3) : 1
RELACION A LISTARSE : A
LEE X INDICE,CLASIF,NORMAL : C
COLUMNA DE CLASIFICACION : 8
  
```

ESTA TODO CORRECTO S/N? S

ERROR!!

Fig. 9.3.1.1.3

En la fig. 9.3.1.1.3 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Este error se presenta cuando en alguno de los datos solicitados por pantalla no se ha respetado los rangos establecidos; así, líneas por página puede ser máximo 80, si se digita más de ese valor se presentará este error.

ERROR!! RELACION NO EXISTE

Cuando la relación que se desea listar no existe. Este error retorna el control al menú principal de POLIREP.

ERROR!! COLUMNA NO EXISTE

Este error se presenta cuando la columna de clasificación no existe en la relación a listarse.

ERROR!! COLUMNA NO ESTA CLASIFICADA

Cuando la columna de clasificación no tiene marca de estar clasificada se presenta este error

9.3.1.2 CREACION DEL FORMATO GENERAL DE 2

RELACIONES.-

Cuando el listado es tipo 2 (listado de 2



relaciones) el usuario debera considerar que la 1era. relacion que defina sera una relacion de lectura secuencial y la 2da. relacion que defina sera una relacion de lectura directa a traves de un indice invertido. De acuerdo a eso se le piden los datos necesarios para efectuar estas lecturas.

**** CREACION DE FORMATO GENERAL ****

```

-----
LINEAS POR PAGINA (1...80) : 66
CARACTERES POR LINEA(1..132): 132
# LINEAS CABECERAS(1....7) : 3
# LINEAS DETALLE (1....3) : 1
RELACION A LISTARSE : A
# COL.CRITERIO RELACION 2 : 8
RELACION A LISTARSE # 2 : B
COLUMNA INDICE RELACION 2 : 4
  
```

ESTA TODO CORRECTO S/N? S

ERROR!!

Fig. 9.3.1.1.3

El dato solicitado '# COL.CRITERIO RELACION 2' implica el criterio bajo el cual se va a hacer la lectura del indice invertido de la relacion 2. El dato 'COLUMNA INDICE RELACION 2', implica la columna bajo la cual se va a hacer la

lectura por índice invertido de la relación 2.

En la fig. 9.3.1.1.3 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Este error se presenta cuando en alguno de los datos solicitados por pantalla no se ha respetado los rangos establecidos; así, líneas por página puede ser máximo 80, si se digita más de ese valor se presentará este error.

ERROR!! RELACION NO EXISTE

Cuando la relación que se haya digitado, cualquiera de ellas no existe.

ERROR!! COLUMNA NO EXISTE

Este error se presenta cuando la columna digitada no existe en la relación correspondiente.

ERROR!! COLUMNA NO TIENE INDICE INVERT.

Cuando la columna de índice no tiene generado un índice invertido para ella se presenta este error.

ESTRAT. CRITERIO <> ESTRAT. RELACION 2

Cuando la estrategia de la columna de criterio no es la misma que la estrategia

de la columna de indice.

9.3.2 CREACION DE DATOS DE CABECERAS.-

Para la creacion de los datos de cabeceras el usuario debe tener ya predefinido su listado.

Para cada linea de cabecera el usuario tiene que determinar el numero de campos por cada linea. Se considera como campo a dos datos de diferentes características. Asi, si en la primera linea de cabecera se tienen los siguientes datos:

FECHA: 23-06-84

Esto se debe considerar como 2 campos diferentes.

'FECHA:' sera un campo tipo C (Constante) y

'23-06-84' sera un campo tipo F (Fecha) que le indicara a POLIREP que debe solicitar al inicio de la ejecucion.

Teniendo en cuenta estos datos, luego de ingresados los datos de formato general se va a presentar la siguiente pantalla.

**** CREACION DE DATOS DE CABECERAS ****

- # LINEA DE CABECERA: 1 - # CAMPOS: 3

ESTA TODO CORRECTO S/N? S

Fig. 9.3.2.1.

Una vez solicitado el # de campos de la linea se le van a ir solicitando al usuario los datos de cada campo. Asi mismo, se le van a ir pidiendo los # de campos de cada linea de cabecera y los datos de cada campo de esa linea.

De acuerdo al tipo de campo que el defina se le van a solicitar los datos. Estos pueden ser de los siguientes tipos:

C : Constante, cuando va a definir un valor constante para la cabecera.

F : Fecha, cuando el desea que en el listado conste la fecha. Esta fecha sera solicitada al inicio de la ejecucion del formato. Solo una vez en todo el formato se le permitira este tipo de campo.

- : Rayado, cuando el usuario desea adornar sus cabeceras define este tipo de campo y al momento de la ejecucion la linea donde ha sido definido este dato se llenara de rayitas.

V : Variable, cuando el usuario define este tipo de campo, debe estar consciente de que implica un quiebre de control a nivel de cabeceras. POLIREP validara este quiebre y saltara de pagina en la impresion al detectarlo. Cuando

el listado es tipo 1, el usuario además tendrá la opción de definir una relación de lectura directa, que servirá para que el usuario saque datos de la misma y mejorar la presentación de las cabeceras.

Tipo C (constante)

```

**  CREACION DE DATOS DE CABECERAS  **
-----
- # DE LINEA CABECERA: 1 - # CAMPO : 1
- TIPO DE CABECERA (C,V,F,-): C
- SALTO ANTES (MAX. 5)      : 3
- POSICION                  : 10
- VALOR: FECHA:

      ESTA TODO CORRECTO S/N? S

      ERROR!! .....
  
```

Fig. 9.3.2.2.

El 'SALTO ANTES' se solicita solo en el primer campo de cada línea para decidir en la impresión cuántas líneas desea dejar en blanco entre cada línea.

En la fig. 9.3.2.2 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Cuando este error se presenta se debe a que la Posición sobrepasa el # de caracteres definido por

línea.

ERROR!! VALOR DE CABECERA MUY LARGO

Cuando la longitud del valor de la cabecera mas la posición definida sobrepasa el # de caracteres definidos por línea.

Tipo F (Fecha)

```

**  CREACION DE DATOS DE CABECERAS  **
-----
- # DE LINEA CABECERA: 1 - # CAMPO : 1
- TIPO DE CABECERA (C,V,F,-): F
- SALTO ANTES (MAX. 5)      : 3
- POSICION                  : 10

      ESTA TODO CORRECTO S/N? S

      ERROR!! .....

```

Fig. 9.3.2.3.

El 'SALTO ANTES' se solicita solo en el primer campo de cada línea para decidir en la impresión cuantos espacios deben dejarse entre cada línea. En la fig. 9.3.2.3 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Cuando este error se presenta se debe a que la Posición sobrepasa el # de caracteres definido por línea.

Tipo - (Rayado)

```

**  CREACION DE DATOS DE CABECERAS  **
-----
- # DE LINEA CABECERA: 1 - # CAMPO : 1
- TIPO DE CABECERA (C,V,F,-): -
- SALTO ANTES (MAX. 5)      : 3

      ESTA TODO CORRECTO S/N? S

```

Fig. 9.3.2.4.

El 'SALTO ANTES' se solicita solo en el primer campo de cada linea para decidir en la impresion cuantas lineas desea dejar en blanco entre cada linea.

Tipo V (variable)

```

**  CREACION DE DATOS DE CABECERAS  **
-----
- # DE LINEA CABECERA: 1 - # CAMPO : 1
- TIPO DE CABECERA (C,V,F,-): V
- COLUMNA A LISTARSE      : 3
- RELACION A LISTARSE     : 2
- COLUMNA A LISTARSE      : 5
- SALTO ANTES (MAX. 5)    : 3
- POSICION                : 10

      ESTA TODO CORRECTO S/N? S

      ERROR!! .....

```

Fig. 9.3.2.5.

En la fig. 9.3.2.5 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Se debe a que no se ha respetado los rangos preestablecidos para cada dato; así, cualquier # de columna no debe sobrepasar 10, pues solo se permiten 10 columnas por relación.

ERROR!! COLUMNA NO EXISTE

Cuando la columna digitada no existe en la relación principal. En el caso de que se de el error en la 2da. columna a listarse, es porque la columna no existe en la relación de lectura directa.

ERROR!! RELACION NO EXISTE

Cuando al digitar el nombre de una relación en el parametro 'RELACION DE LECTURA DIRECTA' y esta no existe, se presenta este error.

ESTRATEGIA DE LA COLUMNA NO ES ENTERA

Cuando se digita una relación para lectura directa, se valida que la columna a listarse # 1 tenga como estrategia 'E', porque de aquí se tomara el # de tupla física para efectuar la lectura directa. Si no es 'E' se presenta este error.

Esta opcion de digitar una relacion de lectura directa y una columna a listarse en esa relacion se la da solo si el listado es tipo 1. Si el usuario en la relacion de lectura directa digita <RETURN> no se pide la columna 2 y no se efectua ninguna lectura directa.

Importante:

En todo lo que respecta a datos a ingresarse en POLIREP, se utiliza un procedimiento predefinido de tal manera que cuando se digita un solo caracter, no se debe presionar la tecla <RETURN>, ya que se recibe el dato inmediatamente y a la vez es validado internamente, y si no es correcto se vuelve a solicitar la digitacion del dato.

9.3.3 CREACION DE DATOS DE DETALLE

Para cada línea de detalle el usuario tiene que determinar el número de campos por cada línea. Se considera como campo a una columna o un resultado de una operación entre 2 columnas.

Al igual que en las cabeceras, se solicitan el # de campos por cada línea. Esta petición del # de campos varía de acuerdo al tipo de listado.

9.3.3.1 CREACION DE DATOS DE DETALLE

(LISTADO TIPO 1).-

Teniendo en cuenta estos datos, luego de ingresados los datos de cabeceras se va a presentar la siguiente pantalla.

**** CREACION DE DATOS DE DETALLE ****

- # LINEA DE DETALLE : 1 - # CAMPOS : 3

ESTA TODO CORRECTO S/N? S

Fig. 9.3.3.1.1

Una vez solicitados el # de campos de la línea se le van a ir requiriendo al

usuario los datos de cada campo. Asi mismo, se le van a ir pidiendo el # de campos de cada linea de detalle y los datos de cada campo de esa linea.

9.3.3.2 CREACION DE DATOS DE DETALLE

(LISTADO TIPO 2).-

Luego de ingresados los datos de cabeceras se va a presentar la siguiente pantalla.

**** CREACION DE DATOS DE DETALLE ****

- RELACION : A - # CAMPOS : 3
- RELACION : B - # CAMPOS : 5

ESTA TODO CORRECTO S/N? S

Fig. 9.3.3.2.1

Una vez solicitados el # de campos de cada relacion, se van a ir solicitando el # de campos de cada linea. Si solo se ha definido una linea de detalle entonces no se hara la peticion de campos por esa linea, pues se asume que sera el resultado de la suma de los campos de la relacion 1 mas el # de campos de la relacion 2.

Si se tiene varias lineas de detalle, la solicitud del numero de campos por cada linea se hara igual que en la fig. 9.3.3.1.1.

9.3.3.3 PROCESO GENERAL

En cualquiera de las 2 pantallas (fig. 9.3.3.1.1 y 9.3.3.2.1) se puede presentar el siguiente error:

ERROR!! NUMERO DE CAMPOS ERRADO

Cuando el # de campos de cada linea sobrepasa a 10 que es el maximo de campos por linea, o en caso de la Fig. 9.3.3.2.1. cuando la suma del # campos de la relacion 1 mas el # de campos de la relacion 2 sobrepasa a 25.

De acuerdo al tipo de campo que el usuario defina se le van a solicitar los datos; asi, puede tener los siguientes tipos de campos:

N : Normal, cuando se va a listar una columna de la relacion.

V : Virtual, el resultado de una operacion cualquiera entre 2 columnas.

Tipo N (normal)

```

**  CREACION DE DATOS DE DETALLE  **
-----
- # DE LINEA DETALLE: 1 - # CAMPO: 1
- RELACION : A          - # CAMPO: 1
-----
- TIPO LINEA DE DETALLE (N,V) : N
- # DE COLUMNA A LISTARSE      : 8
- SALTO ANTES (MAX. 5)         : 3
- POSICION (X,Y)               : 10
- DESEA PROMEDIO S/N           : N
- QUIEBRE/TOTAL/NORMAL (Q,T,N): N
- CRITERIO DE SELECCION: > 10. AND < 11.

      ESTÁ TODO CORRECTO (S/N) ? S .

      ERROR!! .....

```

Fig. 9.3.3.3.1.

La línea donde se detalla el nombre de la relación y el # de campo dentro de esa relación solo se presenta cuando el listado es tipo 2.

El 'SALTO ANTES' se solicita solo en el primer campo de cada línea para decidir en la impresión cuantas líneas desea dejar en blanco entre cada línea.

La 'POSICION (X,Y)' implica la posición inicial y la posición final dentro de la línea. El usuario debe proveer que el

dato no se trunque dentro de la linea de detalle.

En la fig. 9.3.3.3.1. se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Cuando este error se presenta se debe a que la Posicion sobrepasa el # de caracteres definido por linea. o la columna es > a 10.

ERROR!! COLUMNA NO EXISTE

Cuando la columna a listarse no existe en la relacion.

ERROR!! POSIC.1 NO PUEDE SER > POSIC.2

La posicion inicial no puede ser mayor que la posicion final definida.

ERROR!! POSICION > A LONGITUD DE LA LINEA
Cualquiera de las posiciones definidas sobrepasa la longitud de caracteres por linea.

ERROR!! POSICIONES NO DISPONIBLES

Las posiciones disponibles ya fueron utilizadas en otro campo de la misma linea de detalle.

ERROR!! DE SINTAXIS

Este error se produce luego de validada la

línea de criterio de selección si no se ha respetado las reglas de gramática.

Tipo V (virtual)

```

**  CREACION DE DATOS DE DETALLE  **
-----
- # DE LINEA DETALLE: 1  - # CAMPO : 1
- RELACION: A           - # CAMPO : 1
-----
- TIPO LINEA DE DETALLE (N,V) : N
- # DE COLUMNA A LISTARSE      : 8
- DESEA PEDIR CONSTANTE (S/N) : N
- # COLUMNA PARA OPERACION     : 2
- OPERACION A REALIZARSE       : +
- SALTO ANTES (MAX. 5)         : 3
- POSICION (X,Y)               : 10
- DESEA PROMEDIO S/N           : N
- QUIEBRE/TOTAL/NORMAL (Q,T,N): N
- CRITERIO DE SELECCION: > 10. AND < 11.

      ESTA TODO CORRECTO (S/N) ? S

      ERROR!! .....

```

Fig. 9.3.3.3.2

El 'SALTO ANTES' se solicita solo en el primer campo de cada línea para decidir en la impresión cuántas líneas desea dejar en blanco entre cada línea.

Cuando en la opción 'DESEA PEDIR CONSTANTE S/N' el usuario responde 'N', se le solicita la digitación de otra columna de la relación para realizar una operación,

pero si el usuario digita 'S', al momento de la ejecucion se le solicitara un valor constante para efectuar la operacion con la columna de operacion 1. Este valor constante se pedira en ejecucion solo 1 vez al inicio de la ejecucion.

En la fig. 9.3.3.3.2 se pueden presentar los siguientes errores:

ERROR!! VALOR FUERA DE RANGO

Cuando este error se presenta se debe a que la Posicion sobrepasa el # de caracteres definido por linea.

ERROR!! COLUMNA NO EXISTE

Cuando la columna a listarse no existe en la relacion.

ERROR!! POSIC.1 NO PUEDE SER > POSIC.2

La posicion inicial no puede ser mayor que la posicion final definida.

ERROR!! POSICION > A LONGITUD DE LINEA

Cualquiera de las posiciones definidas sobrepasa la longitud de caracteres por linea.

ERROR!! POSICIONES NO DISPONIBLES

Las posiciones disponibles ya fueron utilizadas en otro campo de la misma linea

de detalle.

ERROR!! DE SINTAXIS

Este error se produce luego de validada la línea de criterio de selección si no se ha respetado las reglas de gramática.

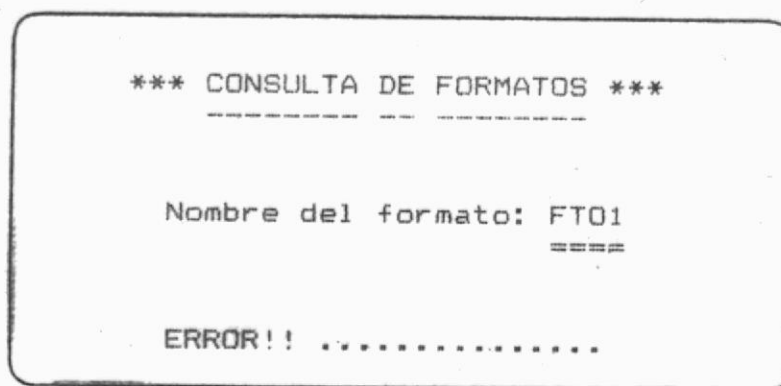
ERROR!! COLUMN DEBE SER ENTERO O REAL

Cuando este error se presenta se debe a que la columna digitada no es de estrategia 'E' o 'R' y si no es así no se puede realizar una operación.

9.4 CONSULTA DE FORMATOS

Esta opción permite consultar cualquier formato generado por el usuario.

Al digitarse la opción '2' del menú principal de POLIREP se le presentará al usuario la siguiente pantalla:



*** CONSULTA DE FORMATOS ***

Nombre del formato: FT01
====

ERROR!!

FIG. 9.4.1.

El usuario deberá digitar el nombre del formato que desea consultar, o presionar <RETURN> si desea regresar al menú principal de POLIREP.

Al momento de definir el nombre del formato pueden presentarse los siguientes errores:

ERROR!! FORMATO DEBE EMPEZAR POR LETRA

El primer carácter de todo nombre de formato debe empezar por una letra.

ERROR!! FORMATO NO EXISTE

Cuando el formato digitado por el usuario no existe.

Si el formato digitado existe entonces se presenta la siguiente pantalla:

*** CONSULTA DE FORMATOS ***

- 1.- CONSULTAR DATOS FORMATO GENERAL
- 2.- CONSULTAR DATOS DE CABECERAS
- 3.- CONSULTAR DATOS DE DETALLE
- 4.- FIN DE LA CONSULTA

DIGITE LA OPCION DESEADA: 1

Fig. 9.4.2



9.4.1 CONSULTA DE FORMATO GENERAL

Para consultar los datos de formato general se tiene que tomar en cuenta el tipo de listado de que se trata. Los datos que se van a presentar dependeran del tipo de listado que se haya definido.

Estos datos se presentan de igual forma como fueron creados. (Ver Numeral 9.3.1).

Luego de presentados los datos se da la opcion de presionar <RETURN> para continuar y se regresa a la pantalla de opciones de CONSULTA (Fig. 9.4.2).

9.4.2 CONSULTA DE DATOS DE CABECERAS

Para consultar los datos de las cabeceras de un formato se pide el # de la línea de cabecera a consultar y el # de campo dentro de la línea de cabecera.

```
*** CONSULTA DE CABECERAS ***  
-----  
  
# LINEA A CONSULTAR : 1  
# CAMPO A CONSULTAR : 3  
  
ESTA TODO CORRECTO S/N? S  
LINEA Y CAMPO NO EXISTEN
```

Fig. 9.4.2.1

Al realizarse la petición de estos datos se buscan en los archivos y si existen se presentan de acuerdo al tipo de campo. Estos datos se presentan de la misma forma que fueron definidos en la creación (ver numeral 9.3.2). Si la línea y campo no existen se presenta el error en pantalla.

9.4.3 CONSULTA DE DATOS DE DETALLE

Para consultar los datos de las líneas de detalle de un formato se pide el # de la línea de detalle a consultar y el # de campo dentro de la línea de detalle.

```
*** CONSULTA DE DETALLE ***  
-----
```

```
# LINEA A CONSULTAR : 1  
# CAMPO A CONSULTAR : 3
```

```
ESTA TODO CORRECTO S/N? S
```

```
LINEA Y CAMPO NO EXISTEN
```

Fig. 9.4.3.1

Al realizarse la petición de estos datos se buscan en los archivos y si existen se presentan de acuerdo al tipo de campo. Estos datos se presentan de la misma forma que fueron definidos en la creación (ver numeral 9.3.3). Si la línea y campo no existen se presenta el error en pantalla.

9.5 CORRECCION DE FORMATOS

Esta opcion permite corregir cualquier formato generado por el usuario.

Al digitarse la opcion '3' del menu principal de POLIREP se le presentara al usuario la siguiente pantalla:

```

*** CORRECCION DE FORMATOS ***
-----

NOMBRE DEL FORMATO: FTD1
=====

ERROR!! .....
  
```

Fig. 9.5.1.

El usuario debera digitar el nombre del formato que desea corregir, o presionar <RETURN> si desea regresar al menu principal de POLIREP.

Al momento de definir el nombre del formato pueden presentarse los siguientes errores:

ERROR!! FORMATO DEBE EMPEZAR POR LETRA

El primer caracter de todo nombre de formato debe empezar por una letra.

ERROR!! FORMATO NO EXISTE

Cuando el formato digitado por el usuario no existe.

Si el formato digitado existe entonces se presenta la siguiente pantalla:

*** CORRECCION DE FORMATOS ***

- 1.- CORREGIR DATOS FORMATO GENERAL
- 2.- CORREGIR DATOS DE CABECERAS
- 3.- CORREGIR DATOS DE DETALLE
- 4.- FIN DE LA CORRECCION

DIGITE LA OPCION DESEADA: 1

Fig. 9.5.2

9.5.1 CORRECCION DE FORMATO GENERAL

Para corregir los datos de formato general se tiene que tomar en cuenta el tipo de listado de que se trata. Los datos que se van a presentar dependeran del tipo de listado que se haya definido.

Estos datos se presentan de igual forma como fueron creados. (Ver Numeral 9.3.1).

Luego de presentados los datos se da la opcion de escoger que dato presentado en la pantalla se desea corregir y se permite cambiar el dato validandolo igual que en la creacion.

Luego de hechas las correcciones deseadas se permite el regreso a la pantalla de opciones de CORRECCION (Fig. 9.5.2).

9.5.2 CORRECCION DE DATOS DE CABECERAS

Para corregir los datos de las cabeceras de un formato se pide el # de la linea de cabecera a corregir y el # de campo dentro de la linea de cabecera.

```
*** CORRECCION DE CABECERAS ***
```

```
# LINEA A CORREGIR : 1  
# CAMPO A CORREGIR : 3
```

```
ESTA TODO CORRECTO S/N? S
```

```
LINEA Y CAMPO NO EXISTEN
```

Fig. 9.5.2.1

Al realizarse la peticion de estos datos se buscan en los archivos y si existen se presentan de acuerdo al tipo de campo. Estos datos se presentan de la misma forma que fueron definidos en la creacion (ver numeral 9.3.2). Si la linea y campo no existen se presenta el error en pantalla. Se permite la correccion de los datos presentados y se valida en igual forma como fueron creados. Luego se regresa a la pantalla de opciones de la CORRECCION.

9.5.3 CORRECCION DE DATOS DE DETALLE

Para corregir los datos de las lineas de detalle de un formato se pide el # de la linea de detalle a corregir y el # de campo dentro de la linea de detalle.

```
*** CORRECCION DE DETALLE ***
-----

# LINEA A CORREGIR : 1
# CAMPO A CORREGIR : 3

ESTA TODO CORRECTO S/N? S
LINEA Y CAMPO NO EXISTEN
```

Fig. 9.5.3.1

Al realizarse la peticion de estos datos se buscan en los archivos y si existen se presentan de acuerdo al tipo de campo. Estos datos se presentan de la misma forma que fueron definidos en la creacion (ver numeral 9.3.3). Si la linea y campo no existen se presenta el error en pantalla. Luego se permite corregir cualquiera de los datos presentados y se validan de la misma forma como se validaron para su creacion.

Luego de hechas todas las correcciones deseadas se

permite retornar a la pantalla de opciones de
CORRECCION.

9.6 EJECUCION DE FORMATOS

Esta opcion permite ejecutar cualquier formato generado por el usuario.

Al digitarse la opcion '4' del menu principal de POLIREP se le presentara al usuario la siguiente pantalla:

```

*** EJECUCION DE FORMATOS ***
-----

NOMBRE DEL FORMATO: FT01
=====

***\ FAVOR PONER PAPEL DE 80 LINEAS /***
***\      POR 132 CARACTERES      /***

-----> CARGANDO LINEAS DE CABECERAS <-----

-----> CARGANDO LINEAS DE DETALLE <-----

FECHA: DDMMAA

CONSTANTE: 10.00

FORMATO EJECUTADO
  
```

Fig. 9.6.1

La FECHA se solicita cuando el usuario ha definido en cabecera algun campo tipo F (Fecha). Esta fecha se imprimira en el lugar del campo tipo F.

La CONSTANTE es solicitada la primera vez que se detecta un campo virtual donde se haya deseado pedir constante

al momento de ejecucion.

El usuario debera digitar el nombre del formato que desea ejecutar, o presionar <RETURN> si desea regresar al menu principal de POLIREP.

Al momento de definir el nombre del formato pueden presentarse los siguientes errores:

ERROR!! FORMATO DEBE EMPEZAR POR LETRA

El primer caracter de todo nombre de formato debe ser una letra.

ERROR!! FORMATO NO EXISTE

Cuando el formato digitado por el usuario no existe.

ERROR!! NUMERO INCORRECTO

Cuando el numero digitado en la CONSTANTE de un campo a listarse, no tiene la misma estrategia del campo con el que se va a realizar la operacion. Si la estrategia del campo es 'E', la constante que se digite debe ser de la misma caracteristica.

ERROR!! FECHA ERRADA

Cuando no se respeta el formato establecido para fecha se presenta este error. El formato para la fecha es (dia/mes/año).

Una vez finalizada la impresion del reporte se presenta el mensaje de FORMATO EJECUTADO.

9.7 ELIMINACION DE FORMATOS

Esta opción permite eliminar cualquier formato generado por el usuario.

Al digitarse la opción '5' del menú principal de POLIREP se le presentará al usuario la siguiente pantalla:

*** ELIMINACION DE FORMATOS ***

NOMBRE DEL FORMATO: FTD1
=====

ERROR!!

Fig. 9.7.1.

El usuario deberá digitar el nombre del formato que desea eliminar, o presionar <RETURN> si desea regresar al menú principal de POLIREP.

Al momento de definir el nombre del formato pueden presentarse los siguientes errores:

ERROR!! FORMATO DEBE EMPEZAR POR LETRA

El primer carácter de todo nombre de formato debe ser una letra.

ERROR!! FORMATO NO EXISTE

Cuando el formato digitado por el usuario no existe.

Si el formato digitado existe entonces se presenta la pantalla con los datos del formato general, para que el usuario se asegure de que ese es el formato a eliminar. Se le pregunta si realmente desea eliminar el formato y si es así se elimina los archivos del disco.

10. EJEMPLO

En este ejemplo se propone demostrar alguna de las habilidades de la Base de Datos POLIREL, haciendo uso de sus modulos de una manera adecuada y donde el proceso de los mismos sea requerido. Ha sido realizado totalmente con la Base de Datos, utilizando sus estructuras para manipular relaciones como se vera mas adelante.

El ejemplo descrito no trata de ser un sistema aplicable en la vida real, es un Rol de Pagos sencillo a manera de ilustracion, el cual sirve de referencia para mostrar las características de la base de datos, empezando con las UTILITARIAS en donde el usuario FORMATEA las relaciones necesarias para el sistema propuesto, el EDITOR en donde se definen dichas relaciones y se ingresan los datos necesarios para crear las relaciones basicas del Sistema de Roles, como son la Relacion Maestro (M), la relacion Transacciones (T), etc.

Un programa en lenguaje PASCAL se encargara de generar la relacion de Rol de Pagos (R) a partir de las relaciones Maestro (M) y Transacciones (T) para obtener las tuplas comunes, realizando los calculos necesarios y luego adicionando una tupla en la relacion de Rol. Merece especial atencion la estructura del programa y la forma como se usan los procedimientos de la UNIDAD POLIREL para crear

una nueva relacion.

Para finalizar, el Generador de Reportes POLIREP se encargara de emitir la liquidacion del Rol respectivo, a partir de la relacion de Rol creada por el programa PASCAL en donde se encuentran todos los datos necesarios para su impresion y una relacion de nombres de departamentos (N), previamente creada. El Rol de pagos se emitira en varios niveles mostrantdo claramente los quiebres de control, con los titulos y totales que cada nivel requiere.

Esperamos que esta pequena demostracion de un Sistema de Rol de Pagos sea, para aquellos estudiantes con mentalidad creativa y espiritu de investigacion, el punto de partida para el desarrollo e implementacion de nuevos sistemas utilizando la Base de Datos Relacional.

10.1 PASOS A SEGUIR PARA OBTENER EL ROL DE PAGOS

1.- Crear las siguientes relaciones para el Sistema de Rol utilizando la utilitaria FORMATED DE ARCHIVOS:

<M> --> Maestro de empleados
<T> --> Transacciones
<N> --> Tabla de nombres
<R> --> Archivo de Rol

2.- Definir las relaciones formateadas en el paso 1 (segun formato dado) e ingresar los datos respectivos a excepcion de la relacion 'R' que solo sera definida. La relacion 'M' y la 'T' tienen sus datos secuenciales por codigo de empleado.

3.- Ejecutar el programa BDRP001 el cual procesara la relacion 'T' contra el Maestro 'M' para encontrar los registros comunes que seran grabados en la relacion 'R' (datos del Rol).

4.- Con el GENERADOR DE REPORTES se ejecuta el formato BDRP, el cual se encargara de emitir el Rol respectivo, clasificado por departamento y por oficina.

(*) Tanto el programa PASCAL como el formato BDRP han sido desarrollados siguiendo las reglas explicadas en este manual para dichos topicos.

10.2 DESCRIPCION DE LAS RELACIONES DEFINIDAS

RELACION 'M' (MAESTRO DE OBREROS)

CAMPOS	ESTRATEGIA	NOMBRE DE CAMPO
1	E	CODIGO
2	S	NOMBRE
3	S	DIRECC
4	S	TELEFO
5	C	ESTADO
6	E	NUIESS
7	E	CARGAS

RELACION 'T' (TRANSACCIONES DEL ROL)

CAMPOS	ESTRATEGIA	NOMBRE DE CAMPO
1	E	CODIGO
2	R	HOTRAB
3	R	RXHORA
4	R	BONIFI
5	R	DESCUE

RELACION 'N' (NOMBRES DEPARTAMENTOS-OFICINAS)

CAMPOS	ESTRATEGIA	NOMBRE DE CAMPO
1	E	NUMDEP
2	S	NOMBDE

RELACION 'R' (ROL DE PAGOS)

CAMPOS	ESTRATEGIA	NOMBRE DE CAMPOS
1	E	DEPART (*)
2	E	OFIC1 (*)
3	E	CODIGO
4	S	NOMBRE
5	R	JORNAL
6	R	HABER
7	R	DEBE

(*) El CODIGO en la relacion Maestro consta de 5 digitos, el primer digito corresponde al departamento, el segundo a la oficina y los tres ultimos al numero de empleado; han sido separados para el manejo de los niveles de control.

10.3 DESCRIPCION DE FORMATOS

Nombre del Formato: BDRP

Lista 1 o 2 Relaciones: 1

FORMATO GENERAL.-

Lineas por pagina: 60

Caracteres por linea: 80

lineas de cabecera: 6

lineas de detalle: 1

Relacion a listarse: R

Lec. por Indice, Clasific., Normal: N

DATOS DE CABECERA.-

- Linea #1: 3 campos

#1 Tipo: C

Salto:

Posicion: 1

Valor: ** POLIREP **

#2 Tipo: C

Posicion: 27

Valor: ROL DE PAGOS

#3 Tipo: C

Posicion: 65

Valor: PAG. :

- Linea #2: 1 campo

Tipo: C

Salto: 1

Posicion: 27

Valor: =====

- Linea #3: 4 campos

#1 Tipo: C

Salto: 2

Posicion: 1

Valor: Departamento:

#2 Tipo: V

Columna: 1

Relacion directa: N

Columna: 2

Posicion: 16

#3 Tipo: C

Posicion: 65

Valor: Fecha:

#4 Tipo: F

Posicion: 73

- Linea #4: 1 campo

Tipo: -

Salto: 2

- Linea #5: 3 campos

#1 Tipo: C

Salto: 1

Posicion: 1

Valor: OFICINA CODIGO

#2 Tipo: C

Posicion: 22

Valor: N O M B R E JORNAL

#3 Tipo: C

Posicion: 47

Valor: BONIF. HABER DESC. NETO

- Linea #6: 1 campo

Tipo: -

Salto: 1

DATOS DE DETALLE.-

- Linea #1: 8 campos

#1 Tipo: N

Columna: 2

Salto: 1

Posicion: 1,5

Promedio: N

Quiebre/Total/Normal: 0

Criterio: -

#2 Tipo: N

Columna: 3

Posicion: 11,15

Promedio: N

Quiebre/Total/Normal: N

Criterio: -

#3 Tipo: N

Columna: 4

Posicion: 19,35

Promedio: N

Quiebre/Total/Normal: N

Criterio: -

#4 Tipo: N

Columna: 5

Posicion: 36,43

Promedio: N

Quiebre/Total/Normal: T

Criterio: -

#5 Tipo: V

Columna: 6

Constante: N

Columna: 5

Operacion: -

Posicion: 45,52

Promedio: N

Quiebre/Total/Normal: T

Criterio: -

#6 Tipo: N

Columna: 6

Posicion: 54,61

Promedio: N

Quiebre/Total/Normal: T

Criterio: -

#7 Tipo: N

Columna: 7

Posicion: 63,70

Promedio: N

Quiebre/Total/Normal: T

Criterio: -

#8 Tipo: V

Columna: 6

Constante: N

Columna: 7

Operacion: -

Posicion: 73,80

Promedio: N

Quiebre/Total/Normal: T

Criterio: -

(**\$+*)

PROGRAM BDFR01;

```
(*****  
(*  
(* PROGRAMA QUE CREA LA RELACION (R) LA CUAL CONTIE- *)  
(* NE LOS DATOS DEL ROL QUE SERAN EMITIDOS EN CON- *)  
(* JUNTO CON EL GENERADOR DE REPORTES "POLIREP". *)  
(*  
(* DESCRIPCION: *)  
(* SE PROCESAN LOS REGISTROS DEL ARCHIVO DE *)  
(* TRANSACCIONES (T) CONTRA LOS REGISTROS *)  
(* DEL MAESTRO DEL ROL (M). EL EMPAREJAMIENT- *)  
(* TO (MATCHING RECORD) DE TUPLAS ES POR CO- *)  
(* DIGO Y POR CADA DUPLITA ESCOGIDA SE GENE- *)  
(* RA UNA TUPLA EN LA RELACION DE ROL LA *)  
(* CUAL CONTENDRA LOS DATOS DEL ROL CALCULA- *)  
(* DOS PREVIAMENTE. *)  
(* F.E.J.L. *)  
(*****)
```

USES POLIREL,COMMON;

VAR

```
REG1, REG2 : FORMATO;  
CODIGO, INTE : INTEGER;  
CH : CHAR;  
WSUELDO, WHABER, WDESC: REAL;
```

PROCEDURE RUTINAERROR;

```
(*****  
(*  
(* PROCEDIMIENTO PARA EL CONTROL DEL "OPCONDICION" *)  
(* DESPUES DE CADA INVOCACION A LOS PROCEDIMIENTOS *)  
(* DE LA UNIDAD. *)  
(* LOS ESTATUS 95 Y 50 SE TOMAN COMO VALIDOS POR - *)  
(* CUANTO LA RELACION NO TIENE TUPLAS AL INICIAR - *)  
(* EL PROGRAMA. *)  
(*  
(* LA SENTENCIA "MULTIFILE('R') " NOS ASEGURA QUE LA *)  
(* RELACION CORRIENTE SEA 'R' SOBRE LA CUAL SE MAN- *)  
(* TENDRA LA INTEGRIDAD DE LOS DATOS. ESTO ES NECE- *)  
(* SARIO EN UN PROCESO MULTIPLE DE RELACIONES. *)  
(* F.E.J.L. *)  
(*****)
```

BEGIN

CASE OPCONCONDICION OF

00,95,50: EXIT (RUTINAERROR);

99,100 : BEGIN

PROMPTAT(15,'FASE 2: ARCHIVO DE ROL GENERADO ...');

PROMPTAT(21,' ** FIN DE PROCESO O.K. **');

MULTIFILE('R');

FINPROCESO;

EXIT(PROGRAM)

END;

END;

PROMPTAT(21,'EL PROCESO SE CANCELA DEBIDO ERROR');

WRITELN;

WRITELN (' EN EL PROGRAMA OPCONCONDICION = ',OPCONCONDICION);

PROMPTAT(23,'** PRESIONE <RETURN> PARA CONTINUAR **');

CH := GETCHAR([CHR(13)]);

EXIT(PROGRAM);

END;

(* PROGRAMA PRINCIPAL *)

BEGIN

(* FORMATO DE TUPLA PARA LEER EL M)AESTRO *)

REG1 [1] := 'ECODIGO';

REG1 [2] := 'SNOMBRE';

REG1 [3] := 'CESTADO';

REG1 [4] := '*****';

(* FORMATO DE TUPLA PARA LA RELACION T)RANSACCIONES *)

REG2 [0] := '*****';

(* PROCESO PRINCIPAL *)

PAGE(OUTPUT);

PROMPTAT(3,' ** ROL DE PAGOS EN PROCESO **');

PROMPTAT(5,' - BDRP01 -');

PROMPTAT(10,'FASE 1: M)AESTRO / T)RANSACCIONES');

PROMPTAT(11,' EJECUTANDOSE ...');

WHILE TRUE DO

BEGIN

(* SE LEE UNA TUPLA DE LA RELACION T)RANSACCIONES *)

LECSECTU('T',REG2);
RUTINAERROR;

WSUELDO := REPOL [2] * REPOL [3];
WHABER := WSUELDO + REPOL [4];
WDESC := REPOL [5];
CODIGO := ENPOL [1];

REPEAT

(* SE BUSCA LA TUPLA CORRESPONDIENTE EN EL M)AESTRO *)

LECSECTU('M',REG1);
RUTINAERROR;
UNTIL ENPOL [1] >= CODIGO;

IF ENPOL [1] > CODIGO THEN
BEGIN

PROMPTAT(21,' REGISTRO NO EXISTE EN MAESTRO');
PROMPTAT(22,' ** PRESIONE <RETURN> ** ');
READLN;

(* SI LA TUPLA DEL M)AESTRO NO TIENE PAREJA SE DEBE *)
(* EMPAREJARLA CON LA SIGUIENTE TUPLA DE T)RANSACC. *)

TUPLANUMERO:=TUPLANUMERO-1;

END
ELSE
BEGIN

(* SE DEFINE EL NUEVO REGISTRO DE ROL *)

ENPOL[1] := (CODIGO DIV 10000);
ENPOL[2] := (CODIGO DIV 1000) - ENPOL[1]*10;
ENPOL[3] := CODIGO MOD 1000;
STPOL[4] := STPOL[2];
REPOL[6] := WHABER;
REPOL[7] := WDESC;
REPOL[5] := WSUELDO;

(* LA NUEVA TUPLA SE ADICIONA EN LA RELACION 'R' *)

ADICIONTU('R');
RUTINAERROR;
END;

END;

END.

** POLIREP **

ROL DE PAGOS

PAG. :

DEPARTAMENTO: DESARROLLO DE SISTEMAS

FECHA: 16-06-

OFICINA	CODIGO	N O M B R E	JORNAL	BONIF.	HABER	DESC.	NETO
2	310	VERONICA CUCALON	28000.00	2000.00	30000.00	1500.00	28500.
	320	ANA COX	46800.00	3500.00	50300.00	500.00	49800.
	330	GISSELLE SALMON	24000.00	600.00	24600.00	3000.00	21600.
	340	MATILDE PERE	27000.00	2500.00	29500.00	900.00	28600.
SUBTOTALES >>>			125800.00	8600.00	134400.00	5900.00	128500.
4	300	FERNANDO PAEZ	13300.00	1200.00	14500.00	1000.00	13500.
	350	FAUSTO JACOME	21000.00	1800.00	22800.00	800.00	22000.
	360	AGUSTIN CHUSAN	42500.00	1800.00	44300.00	1100.00	43200.
SUBTOTALES >>>			76800.00	4800.00	81600.00	2900.00	78700.
SUBTOTALES >>>>			202600.00	13400.00	216000.00	8800.00	207200.

OLIREP **

ROL DE PAGOS

PAG. :

2

DEPARTAMENTO: ADMINISTRACION

FECHA: 16-06-84

CODIGO	NOMBRE	JORNAL	BONIF.	HABER	DESC.	NETO
1	348 ALFREDO ALVAREZ	26800.00	1500.00	28300.00	1200.00	27100.00
	350 ANDRES NAVARRO	33000.00	1500.00	34500.00	2500.00	32000.00
	355 XAVIER SALINAS	31500.00	2000.00	33500.00	2000.00	31500.00
TOTALES >>>		91300.00	5000.00	96300.00	5700.00	90600.00
5	325 NELSON ESCOBAR	40000.00	2000.00	42000.00	1500.00	40500.00
	346 PABLO CHANG JO	34650.00	2050.00	36700.00	180.00	36520.00
	352 LUIS SANCHEZ	40000.00	4000.00	44000.00	3000.00	41000.00
	370 ISIDRO GUTIERREZ	40000.00	2000.00	42000.00	1200.00	40800.00
TOTALES >>>		154650.00	10050.00	164700.00	5880.00	158820.00
TOTALES >>>>		245950.00	15050.00	261000.00	11580.00	249420.00



** POLIREP **

ROL DE PAGOS

PAG. :

DEPARTAMENTO: PERSONAL

FECHA: 16-06-

OFICINA	CODIGO	N O M B R E	JORNAL	BONIF.	HABER	DESC.	NETO
0	100	MERCEDES VILLACRE	32850.00	1200.00	34050.00	400.00	33650.
	110	MARISOL ARTEAGA	23100.00	3500.00	26600.00	1800.00	24800.
SUBTOTALES >>>			55950.00	4700.00	60650.00	2200.00	58450.
2	120	JULIO CHANG	2200.00	2000.00	4200.00	1200.00	3000.
	140	XAVIER CARDENAS	17400.00	3000.00	20400.00	800.00	19600.
	160	REINALDO ROCA	22500.00	1200.00	23700.00	600.00	23100.
SUBTOTALES >>>			42100.00	6200.00	48300.00	2600.00	45700.
SUBTOTALES >>>>			98050.00	10900.00	108950.00	4800.00	104150.
TOTALES GENERALES >>>			546600.00	39350.00	585950.00	25180.00	560770.

11. RECOMENDACIONES

- Antes de usar el Sistema de Base de Datos POLIREL lea detenidamente el Manual Operativo del mismo.
- No trate de grabar información en los diskettes que contienen el Sistema de Base de Datos (Base 1 y Base 2 y Base 3).
- Trate en lo posible de mantener los archivos con los que está trabajando en un solo diskette.

12.

APENDICES

A) TABLA DE ERRORES

- 1 ** FALTA NOMBRE DE ESTRUCTURA COMANDO **
- 2 ** NOMBRE DE ESTRUCTURA ESPERADO **
- 3 ** NOMBRE DE ESTRUCTURA INCORRECTO **
- 4 ** := ESPERADO **
- 5 ** OPERADOR (.,*,%) ESPERADO **
- 6 ** NOMBRE DE COLUMNA INCORRECTO **
- 7 ** OPERADOR RELACIONAL (.,*,%) ESPERADO **
- 8 ** NOMBRE DE COLUMNA ESPERADO **
- 9 ** ARGUMENTO PARA SELECCION ESPERADO **
- 10 ** COLUMNA(S) PARA PROYECCION INCORRECTA(S) **
- 11 ** EXPRESION INCORRECTA **
- 12 ** NOMBRE ESTRUCTURA NO PERMITIDA EN SELECCION **
- 13 ** ESTRUCTURA PERMANENTE INCORRECTA **
- 14 ** DELIMITADOR (/) ESPERADO **
- 20 ** NO EXISTE COLUMNA EN RELACION **
- 21 ** ESTRATEGIA NO VALIDA **
- 22 ** NOMBRE PERMANENTE EN PROYECCION NO VALIDO **
- 23 ** VARIABLE PERMANENTE EN PROYECCION NO EXISTE **
- 24 ** VARIABLE PERMANENTE EN LIST/DISP NO EXISTE **
- 25 ** VARIABLE PERMANENTE DUPLICADA **
- 26 ** NOMBRE PERMANENTE EN SELECCION NO EXISTE **
- 27 ** OPERADOR #1 DE JOIN NO VALIDO/NO EXISTE **
- 28 ** OPERADOR #2 DE JOIN NO VALIDO/NO EXISTE **

29 ** ESTRUCTURAS FUENTES JOIN INDENTICAS **
30 ** NO EXISTE COLUMNA COMUN EN JOIN **
31 ** HAY MAS DE UNA COLUMNA COMUN EN JOIN **
32 ** NOMBRE PERMANENTE DE JOIN NO EXISTE **
33 ** OPERACION PARA JOIN NO PERMITIDA **
34 ** NOMBRE PERMANENTE DE JOIN NO VALIDO **
35 ** NO EXISTE SELECCION EN TABLA DE SELECCIONES **
36 ** NO EXISTE JOIN EN TABLA DE JOINS **
37 ** NO EXISTE PROYECCION EN TABLA DE PROYECCIONES **
38 ** NUMERO ENTERO EXCEDE A 5 DIGITOS **
39 ** NUMERO REAL INCORRECTO **
40 ** TABLA GENERAL DE PROCESOS LLENA **
41 ** NO EXISTE CONSULTA CATALOGADA **
42 ** COLUMNAS REPETIDAS EN PROYECCION **
43 ** NO EXISTE TUPLAS EN JOIN **
44 ** NO EXISTE TUPLAS EN SELECCION **
45 ** TABLA DE PROYECCIONES LLENA **
50 ** RELACION NO TIENE TUPLAS **
51 ** RELACION NO TIENE COLUMNAS **
75 ** RELACION ESTA ELIMINADA **
76 ** RELACION NO ESTA DEFINIDA **
91 ** UNO DE LOS ARCHIVOS NO EXISTE **
94 ** NO EXISTE NOMBRE DE COLUMNA **
99 ** FIN DE ARCHIVOS **
101 ** NUMERO DE ARCHIVOS EXCEDE AL MAXIMO **

B) TABLA DE CONDICION DE OPERACION

En este apendice se describe el significado de los posibles valores que puede tomar la variable OFCONDICION al utilizar los procedimientos de la unidad Polirel.

VALOR	SIGNIFICADO
00	Operacion exitosa
21	Valor identificador de la clave perdido en el procedimiento LECDIRTU.
22	Numero de columna dado para nombre de archivo no valido (LECDIRTU).
26	No existe archivo invertido para dicha relacion o no encuentra informacion de control en el archivo ADT.
27	No existe tuplas para dicho criterio de seleccion.
50	Numero de tuplas en relacion igual a cero.
51	Numero de columnas en relacion igual a cero.
60	Longitud de atributo a insertar es igual a cero.
70	Numero de registro a eliminar no esta dentro del rango permitido.
71	Invoca al procedimiento ELIMINTU sin haber leído una tupla previamente.

- 75 Relacion eliminada.
- 76 La relacion ha sido solo formateada y no esta definida.
- 80 Numero de registro a actualizar no esta dentro del rango permitido.
- 81 Trata de actualizar una tupla sin haber leído una previamente.
- 82 Atributo tipo string tiene longitud cero.
- 91 Archivo no existe o un error ha ocurrido al tratar de accesarlo.
- 92 Nombre de relacion no permitido por Polirel
- 94 Uno de los atributos en el registro definido por el usuario no valido.
- 95 Valor de parametro igual a cero en procedimiento para encontrar la tupla fisica.
- 99 Fin de datos en la lectura secuencial de una relacion (LECSECTU).
- 100 Fin de datos para archivos invertidos y/o de clasificacion.
- 101 Trata de leer mas de tres archivos concurrentemente (inicialice el sistema)
- 102 Error al tratar de accesar al archivo de clasificacion o invertido; el archivo no ha sido encontrado.

C) PALABRAS RESERVADAS POR LA UNIDAD POLIREL

En este apéndice se describen todas aquellas palabras que son utilizadas por la Unidad Polirel, por lo tanto tienen el carácter de reservadas y NO deben ser utilizadas en los programas del usuario para evitarse resultados inesperados.

CONSTANTES:

- cero
- maxcolumnas
- tuplelength
- vacío
- zero

TIPOS:

- formato
- setofchar
- tbuffers
- tarchivos

VARIABLES:

adf	numadfbuffes
adfbuffer	numcolumnas
adfcounter	numtfbuffers
alfabeto	numeros
bp01	numtuplas
bp02	opcondicion
bp03	pc02
bp04	ppc
capol	pradf
countadf	prtdf
counttf	prtf
dtfcounter	registro
enpol	repol
findata	stadic
index	stpol
indexbuffer	tdf
longtupla	tdfbuffer
mfiles	tf
modos	tfbuffer
nextbucked	tfcounter
nextposic	tuplalogica
nextrecord	tuplanumero

PROCEDIMIENTOS:

actualtu:	Actualiza tuplas.
adiciotu:	Adiciona tuplas.
datocomun:	Obtiene la informacion general sobre una relacion.
elimintu:	Elimina tuplas.
enterostring:	Convierte un entero a string.
finproceso:	Finaliza una operacion sobre una relacion dada.
inicializacion:	Inicializa variables y buffers de trabajo.
lecdirtu:	Lee en forma directa una tupla.
lecsectu:	Lee en forma secuencial las tuplas.
llenoblanco:	Inicializa una variable string con N blancos.
multifile:	Permite la manipulacion concurrentemente de hasta 3 relaciones.
readfbuffer:	Lee un bloque del archivo ADA.
readtfbuffer:	Lee un bloque del archivo AT.
readfisc:	Obtiene los datos de una tupla fisica dada.
realstring:	Convierte un numero real a string.
retirodato:	Obtiene el dato para una columna especifica.
retirocolumna:	Obtiene los datos de una columna.

stringentero: Convierte un string a su correspondiente valor entero.

stringreal: Convierte un string a su correspondiente valor real.

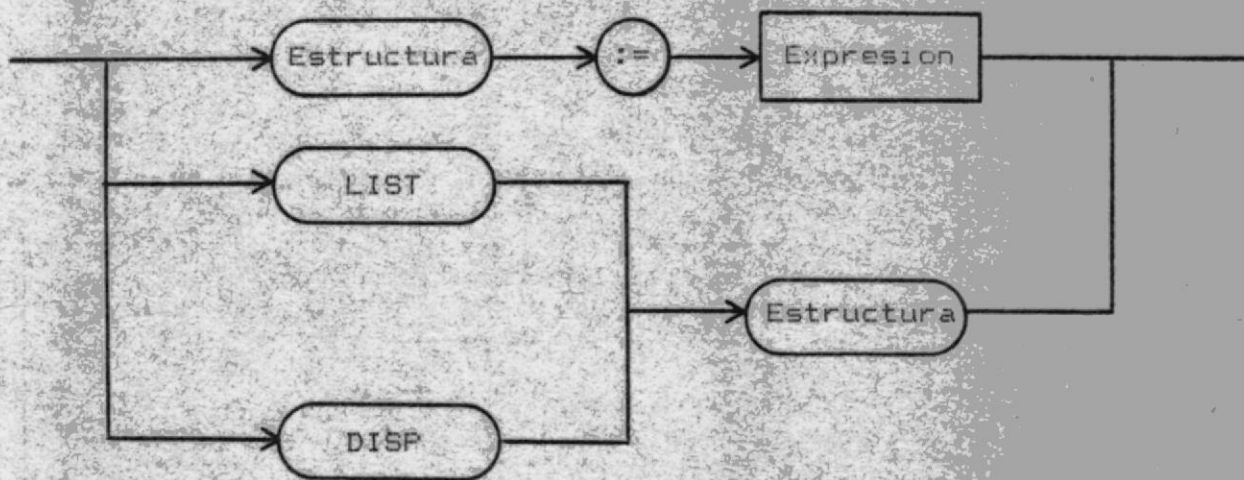
tuplafisica: Obtiene el numero de tupla fisica dada una tupla logica.

validarchivos: Verifica que existan los archivos de una relacion.

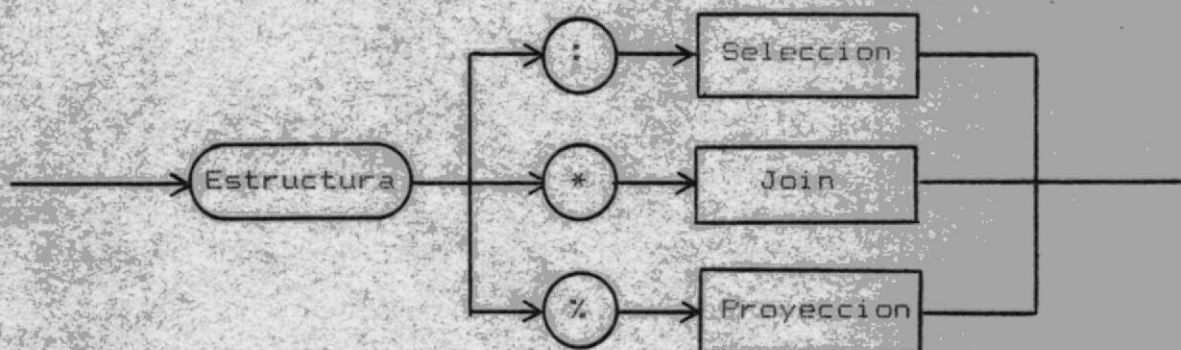
validname: Valida el nombre de una relacion.

D) DIAGRAMAS SINTACTICOS

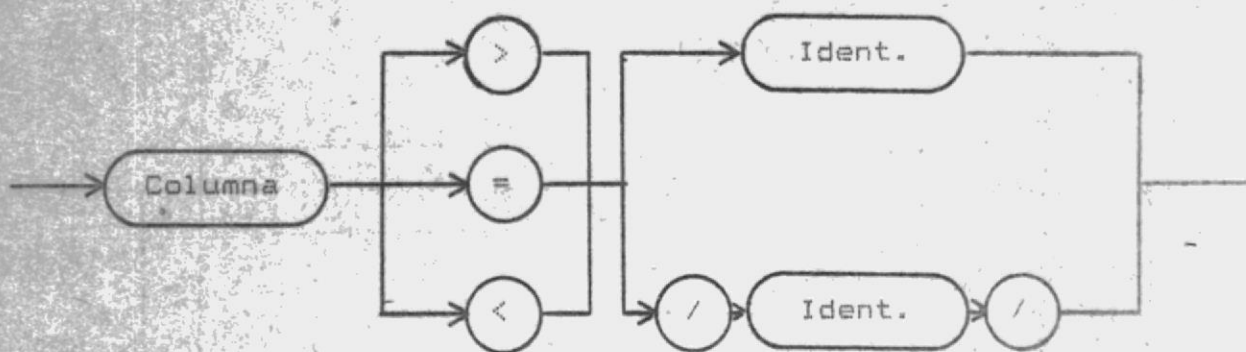
Sentencia



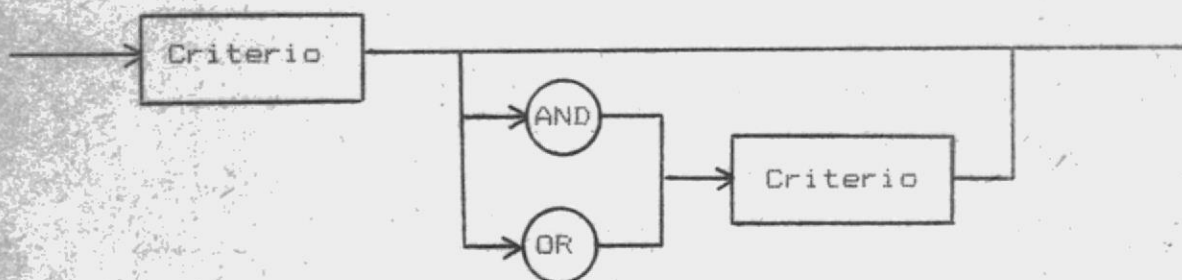
Expresion



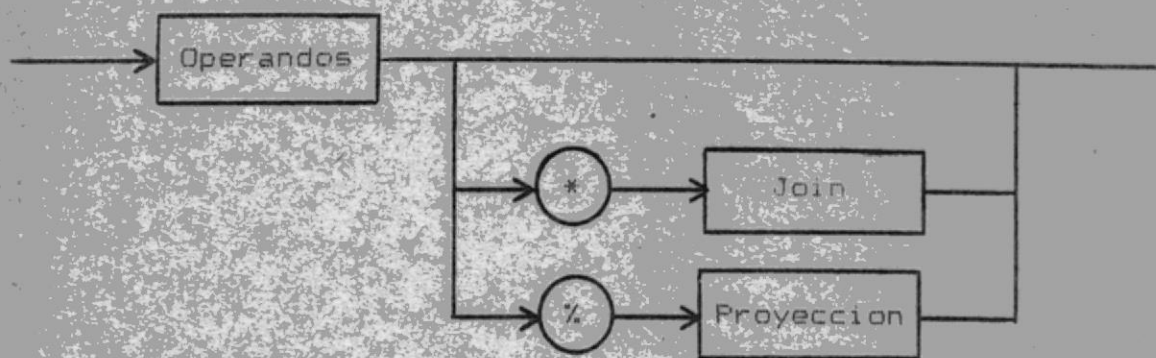
Criterio



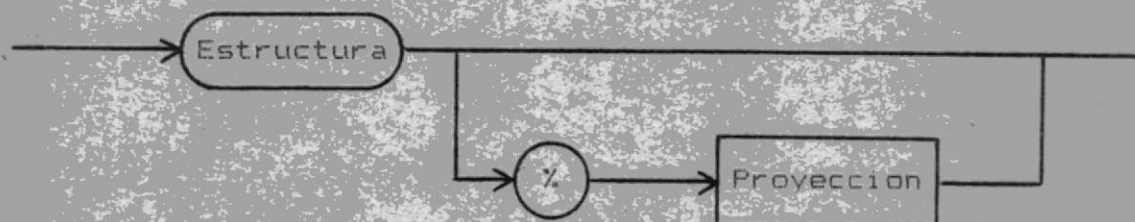
Operandos



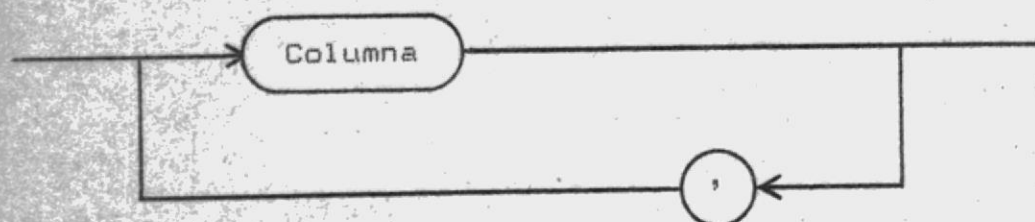
Selección



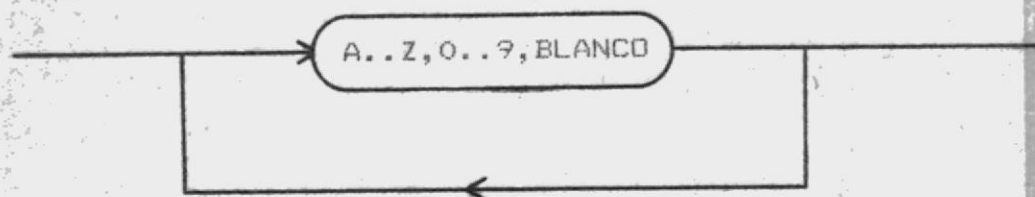
Join



Proyeccion



Identificador



Observacion:

En el criterio de seleccion para aquellas columnas cuya estrategia sea STRING, el valor del identificador debe ir entre "/".

SA := A : NOMBRE = /FERNANDO PAEZ/